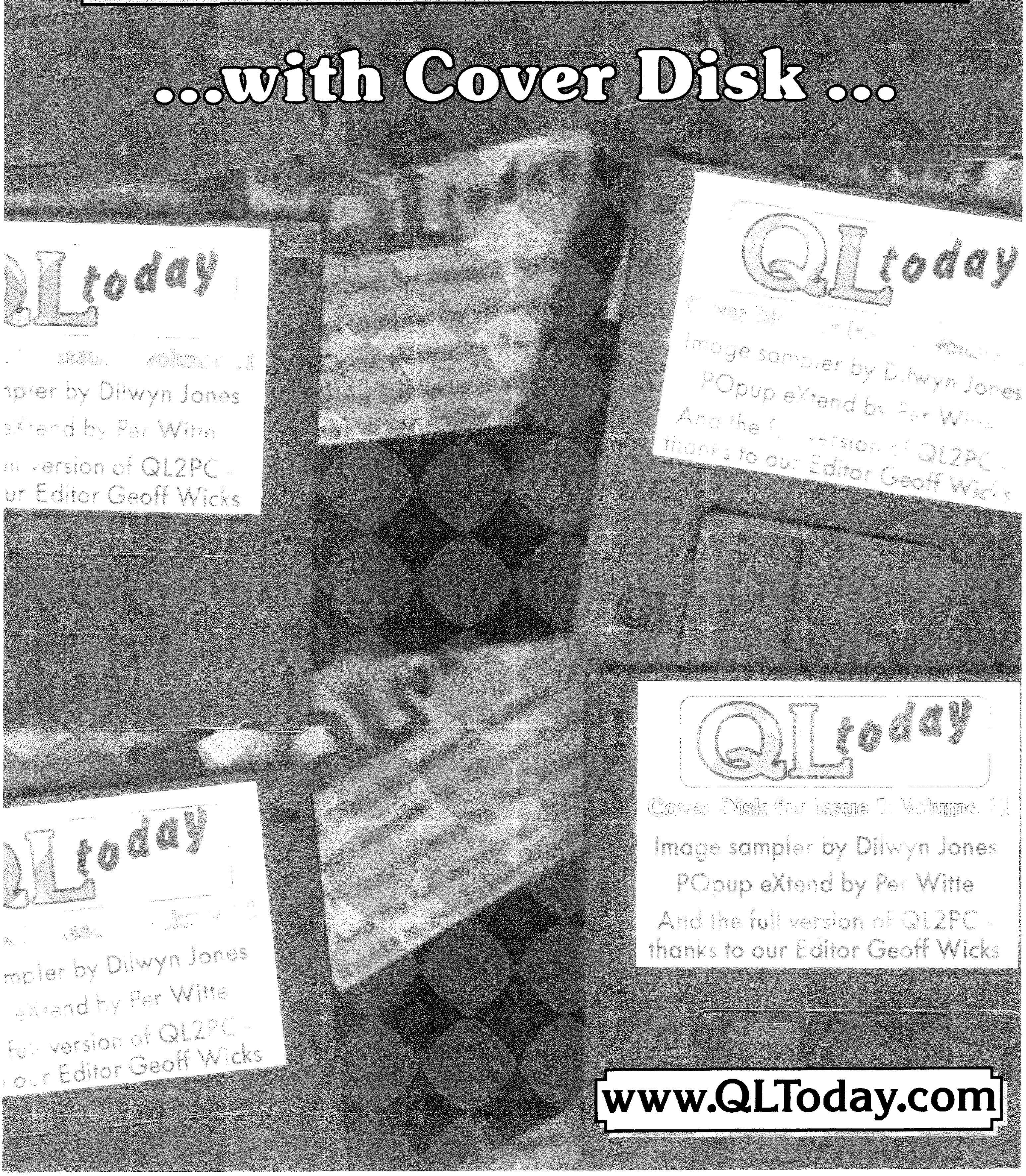
QL Today

Volume 11
Issue 2
Sept.-November
2006

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...

...with Cover Disk ...


QL today
Cover Disk for Issue 2 Volume 11
Image sampler by Dilwyn Jones
POpup eXtend by Per Witte
And the full version of QL2PC -
thanks to our Editor Geoff Wicks

www.QLToday.com

Contents

Advertisers
in alphabetical order

QL *Today*

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software	Tel. +49 203 502011
Kaiser-Wilhelm-Str. 302	Fax +49 203 502012
47169 Duisburg	email: JMerz@j-m-s.com
Germany	email: QLToday@j-m-s.com

English office:

Q Branch	Tel. +44 1273 386030
20 Locks Hill	Mobile +44 7836 745501
Portslade	Fax +44 1273 381577
BN41 2LB	email: qbranch@qbranch.demon.co.uk
United Kingdom	email: QLToday@j-m-s.com

Editor:

Geoff Wicks	Tel. +44 1332 271366
5b Wordsworth Avenue	email: gwicks@beeb.net
Sinfin	email: QLToday@j-m-s.com
Derby DE24 9HQ	
United Kingdom	

Co-Editor:

Bruce Nicholls	Tel +44 20 71930539
38 Derham Gardens	Fax +44 870 0568755
Upminster	email: qltoday@q-v-d.demon.co.uk
Essex RM14 3HA	email: QLToday@j-m-s.com
United Kingdom	

QL *Today* is published five times a year, our volume begins on beginning of June. Please contact the German or English office for current subscription rates or visit our homepage www.QLTODAY.com.

We welcome your comments, suggestions and articles. YOU make QL *Today* possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

If you need more information about the UNZIP program which is used by our BOOT program to unpack the files, we suggest that you visit Jonathan Hudsons web site where you find more information about lots of interesting QDOS software and INFOZIP at www.bigfoot.com/~jrhudson/

The deadline for the next issue is the 20th of December 2006

Editorial

by Geoff Wicks

Is there some evil person who has cast a spell on QL-Today?

I ask this question because earlier this year your editor emulated his predecessor by moaning about his PC, which had slowed to a snail's pace.

The problem was easy to diagnose. The virus software had outgrown the memory on the computer and it was a simple question of adding extra RAM. Well, not quite. The local firm, who had done an excellent job in building the machine, were reluctant to do the upgrade. The message was clear. This was to be a do-it-yourself job.

Fitting memory to a PC requires no great ability even for someone with limited hardware skills. It is a lot simpler than many QL hardware jobs I have done over the years. The big problem is knowing what memory to buy. Until I was confronted with this problem, the innards of my PC had not concerned me greatly. Suddenly I had to check out numerous different memory cards and choose the right one for my machine.

This experience was a valuable object lesson for your editor. One of the most frequent comments about QL-Today is over its, at times, high technical content.

Two people spoke to me at the Hove show about this. One said he had never subscribed to QL Today, because it was too difficult for him, the other that he found the last issue an easier read than usual. (As it happens that issue had more general news items than most.)

QL-Today has to cater for a wide range of readership and getting the balance right between highly technical and non-technical content is not easy. If we do our job well each reader will find something to interest him in each issue, but also some articles he does not fully understand. We should avoid putting people off exploring new avenues of QL use by making them appear more complicated than they are. You do not need a technical knowledge of the pointer environment to run PE programs, but you do need it if you have programming ambitions. Similarly it can be fun to experiment with GD2 colours even when you have limited programming skills.

Looking back on my PC problem, I could easily have panicked. There was a huge learning curve, but, now I have done it, I realise it was much simpler than I expected. It would have been easier if I had had more understanding from the native PC experts. After all, who did I turn to when I needed expert and reliable advice? Not the PC experts but QL traders.

A word from the publisher...

by Jochen Merz

You may have notice print quality changes during the last few issues.

We ran some tests of producing the magazine electronically, using PDF. The results were poor both times (last issues of Volume 10). Issue 1 of Volume 11 was done in the old-fashioned way (master print), and the result is very good again. While doing the PDF test, it became clear, that there will never be an electronic version of QL Today. As the magazine is produced on Calamus, an ATARI program running under Windows, the PDF does not use fonts but bitmaps only, resulting in enormously large files, even at resolutions like 150 or 300dpi. A whole issue in greyscale wouldn't even fit on a CD, it requires a DVD (and it takes over 24 hours on a fast machine to generate).

Sorry about doing "live tests", initial test looked reasonably OK, but it was then done produced under time pressure. No tests anymore, we keep the high quality.

Unfortunately, I can't bring this issue personally to Byfleet as planned ... but as you will see, it was worth waiting for ... loads of interesting articles and a bonus disk containing a full version of a very useful program (kind of early Xmas present)... enjoy!

New Website

Just Words! have launched a completely re-written website with more download possibilities. From the home page you now have a choice of four pages:

DOWNLOADS allows you to download the Just Words! freeware and commercial software plus a couple of other programs.

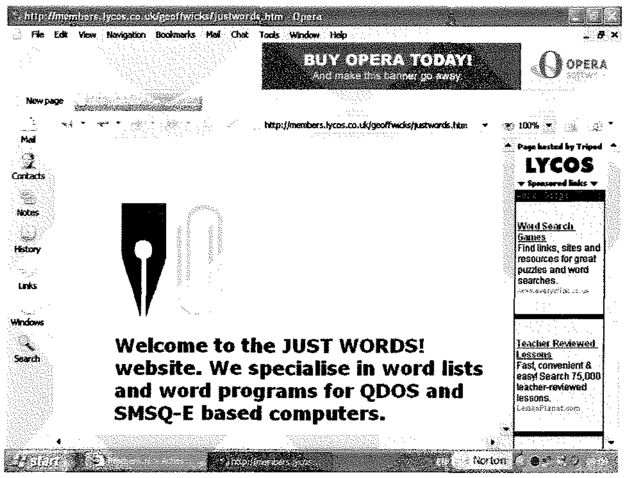
WORD LISTS contains the complete Just Words! range of QTYP dictionaries as well as the Vocabulary Database and QTYP expand utility.

ADVICE has a number of short articles giving answers to the most frequently asked transfer questions.

SHOWS gives up to date details of the QL shows programme.

Just Words! informs us that the new site is the first stage in a reshaping of the software house to bring it in line with current trends in the QL community. They hope to have further details at the Byfleet show.

http://members.lycos.co.uk/geoffwicks/justwords.htm



http://members.lycos.co.uk/geoffwicks/justwords.htm - Opera

File Edit View Navigation Bookmarks Mail Chat Tools Window Help

BUY OPERA TODAY! And make this banner go away.

Welcome to the JUST WORDS! website. We specialise in word lists and word programs for QDOS and SMSQ-E based computers.

Page hosted by Tripod LYCOS

Word Search Games
Find links, sites and resources for great puzzles and word searches.

Teacher Reviewed Lessons
Fast, convenient & easy! Search 75,000 teacher-reviewed lessons.
LessonPlanet.com

QUANTA Changes

There have also been changes at the Quanta website although these are largely invisible to the user. In the June/July 2006 edition of the Quanta Magazine John Gilpin writes that Quanta's old host, Ghoulnet, closed their service on 31st May 2006. The website is now hosted by the Continum Group who successfully transferred the site and the Quanta email service to the new servers within 24 hours. The only difference the user will notice is the Continum logo on the web pages.

Quanta has long had an ambition to improve the quality and reliability of its website, but so far has been unable to find a volunteer to act as webmaster.

www.quanta.org.uk

Graphics Developments

In recent months there have been interesting developments in graphics programs for transferring images between the QL and the PC.

At the beginning of August Dilwyn Jones invited subscribers to the QL-users email group to test the alpha version of a conversion program he was writing. In his own words:

"I've put an alpha test release of my QL/Windows BMP graphics conversion and viewing program onto the Graphics page on my website.

This is the first release version and no doubt subject to the usual bugs and problems of first releases.

The program lets you convert to and from 24-bit Windows BMP graphics files, and handles QL mode 4, mode 8, mode 16 (Aurora 256 colour), mode 32 (QPC2/QXL) and mode 33 (Q40/Q60) graphics files. BMP can also convert between the various QL modes. Where relevant, it can also convert Page Designer pages (e.g. transfer them to Windows BMP format).

File viewing is implemented across modes. For example, if you are using mode 32 on QPC, and wish to view mode 4, mode 8, mode 16 or mode 33 pictures, the program will attempt to convert and display in your current screen mode. Results vary - trying to view a 24-bit BMP or 16-bit GD2 image in mode 4 will inevitably give poor results.

The program needs expanded memory and a system with Window Manager 2 (either recent versions of SMSQ/E or QDOS with pointer environment version 2).

This is a first release, so no doubt it will suffer from the usual bugs and weaknesses of first releases. Feedback is invited so that I may try to improve the program, as it is intended to be the front end viewer program for my wallpaper images CD/DVD which is ready once I'm happy with this program."

Several members of the user's group responded favourably to this invitation, but at the time of writing the program was still being developed. The version currently on Dilwyn's website is a beta version.

www.dilwyn.uk6.net/graphics/index.html

George Gwilt has also continued to work on his screen snatcher suite, SVSCR, improving its user friendliness, compatibility with most QL systems, on-screen information and adding _pic format support. George has also been upgrading some of his other programs, and the SQLUG website, from which his software can be downloaded, now has a version of unzip that does not need unzipping itself. Details are in the next news item.

George Gwilt Upgrades

1. The extensions TurboPTR, TPTR_EXT, have been upgraded to allow a quicker method of finding the maximum screen size. This is needed for SMSQE v3.12 and later. Also an error in SET_PTR, which sets the pointer to a particular position, has been corrected. This error affected programs run under non SMSQE operating systems.
2. As a consequence of the changes in 1, the programs SVSCR, NET_PEEK, GWDISS and DISP have all been upgraded.
3. A simple means of compiling SMSQE v3.12 using GWASS is on the SQLUG site.
4. The program "unzip" can be downloaded and used from the SQLUG site without having to be unzipped!

http://www.jms1.supanet.com

Q-emuLator Upgrade

Daniele Terdina writes:

"A new version of Q-emuLator for Windows is available. The main new feature is emulation of the Q60 video modes and some of the Aurora video modes.
RAM available to unregistered copies has also been increased to 384KB to allow users to run unzip_exe to expand archives of QL software downloaded from the Internet.
You can download Q-emuLator 2.4 from
http://users.infoconex.com/~daniele/winql.html"
In reply to queries about accessing the newly implemented modes Daniele wrote:
"Needs either SMSQ/E for the Super Gold Card (it now includes Aurora colour drivers) or QL software that directly accesses the video hardware. SMSQ/E for Q40/Q60 may also work (and give you more colours and resolution than the SGC version) but I have not tested it."
Daniele further stressed the importance of having the correct operating system for accessing the new modes. In reply to a person attempting to emulate Q60 mode from SMSQ/E for the Gold Card:
"That wouldn't work: SMSQ/E for the Gold Card doesn't have Q60 video drivers. Select Aurora instead of Q60, and make sure you use the version of SMSQ/E with the Aurora GD2 video drivers.
Q-emuLator emulates the Aurora and Q60 video cards, but if you just use Sinclair or Minerva ROM images, these versions of QDOS don't know how to draw to the extended graphics modes. You need to either use QL software that directly accesses the extended video memory, or a version of the OS that does the same, like SMSQ/E with the GD2 drivers for the video card you wish to use."

SHELL v. 1.14

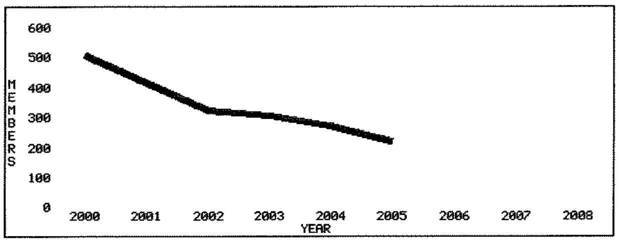
Malcolm Cadman writes:

"Shell version 1.14 is now available from the London QL and Quanta Group web page at -
http://www.mcad.demon.co.uk/lquan.htm
The Shell program was originally developed for QL by Adrian Ives. This version - now at v1.14 - has been enhanced and includes several bug fixes by David Gilham of the London Group.
What does the Shell do? Well, it is a Unix like command line interpreter, giving direct access to program launching, QL pipes, etc. It is almost the direct opposite of the graphical user interfaces for the QL, like QPAC2 and QDT, well worth having in your suite of QL software for the times that you may need to make use of it.
It is now compatible with the larger QL screens too thanks to a new command.
Please direct any queries to David at the email address below.
David Gilham - **david@davidgilham.me.uk**"

Constitution Update

In the last edition of QL Today we reported an error in the published Quanta constitution. This claims that the signatures of 100 members are required to call a Special General Meeting whereas those of only 5% of the membership (currently about 11) are required. Since our report the silence from Quanta has been deafening.
QL Today now has a copy of the minutes of the 1998 AGM in which the change was made. We can now confirm that the motion to reduce the number of signatures required to call a SGM was passed by 65 votes with none against. One person abstained from voting.
There is an interesting twist to this story. Since 2000 Quanta membership has fallen from 504 to 219. Our graph shows the Quanta membership for each year since 2000 (except 2001). The fall in membership is not flattening out, but, if anything, has begun to increase again. If the Quanta

committee are unable to arrest this decline, then membership will be below 100 by the time their 3 year period of office is completed. According to the constitution Quanta has sent to its members it will then be physically impossible for members to call for an SGM.



Whatever happens, Quanta at the end of the 3 year term of office is likely to be a very different organisation from what it is now. There could be many implications for Quanta finances, the Quanta Magazine and Quanta sponsored shows. The committee have an unenviable task before them, and it would be interesting to hear their plans for the future.

In Praise of Dilwyn

QL Today's editor does not always get his predictions right, but one has been spot on. In his very first editorial he made this prediction about his predecessor:

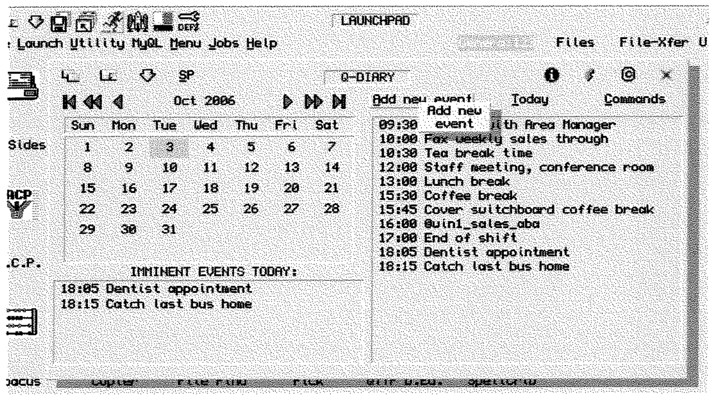
"Just a final word about Dilwyn. I predict that QL Today's loss will prove to be QL programming's gain."

Since leaving QL Today the Welsh wizard has spewed out a steady stream of software, which, since the summer, has almost become a torrent. We had just finished writing our main news section including details of his BMP program, when he sent us news of two more programs and one upgrade.

Dilwyn writes of his new programs:

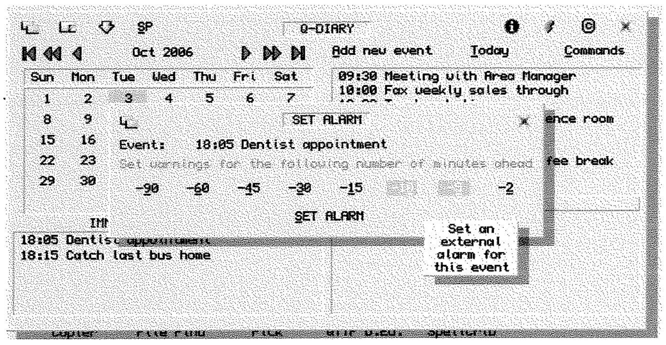
"Q-DIARY and ALARM

Q-Diary is a personal diary program, which lets you schedule events for given dates and times in a similar way to a paper diary.



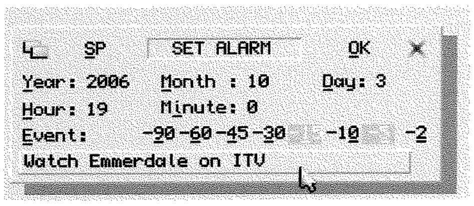
```
Sun Mon Tue Wed Thu Fri Sat
  1   2   3   4   5   6   7
  8   9  10  11  12  13  14
 15  16  17  18  19  20  21
 22  23  24  25  26  27  28
 29  30  31

09:30 Meeting with Area Manager
10:00 Fax weekly sales through
10:30 Tea break time
12:00 Staff meeting, conference room
13:00 Lunch break
15:30 Coffee break
15:45 Cover switchboard coffee break
16:00 @win1_sales_aba
17:00 End of shift
18:05 Dentist appointment
18:15 Catch last bus home

IMMINENT EVENTS TODAY:
18:05 Dentist appointment
18:15 Catch last bus home
```

Events with a date, time and short message are entered into a diary database, with the program displaying daily diary pages via a calendar-style display. Click on the date of the page you wish to display. Facilities are included to display "imminent events" (e.g. a list of diary entries due within the next 15 minutes), and to set alarms to go off (with up to 8 optional warnings ahead of time) when the diary entry is due. Q-Diary has the facility to activate external alarm programs, allowing you to write your own programs to handle diary entries as they fall due.



```
09:30 Meeting with Area Manager
10:00 Fax weekly sales through

Event:    18:05 Dentist appointment
Set warnings for the following number of minutes ahead
  -90   -60   -45   -30   -15         -2

18:05 Dentist appointment
18:15 Catch last bus home
```

Alarm is a program which may be used with Q-Diary or used independently. It can handle the alarms from Q-Diary, or it can let you enter a time and description of some event to happen today and when that time falls due it can either issue an alert message for that event, or even execute a program at that time. Up to 8 advance warnings can be set at periods from 2 minutes ahead up to 90 minutes ahead.



```
Year: 2006   Month : 10   Day: 3
Hour: 19     Minute: 0
Event:     -90-60-45-30    -10    -2
Watch Emmerdale on ITV
```

Both programs are pointer driven and require Window Manager 2 (i.e. SMSQ/E or QDOS with pointer environment version 2 or later).

Both programs are available from my website, at **www.dilwyn.uk6.net/utils/index.html**"

The upgraded program is a "blast from the past".

"BASIC REPORTER UPDATE

Version 2 of my Basic Reporter program is now available from my website. This was the first QL program I sold through DJC! It is now about 16 years old and showing its age, so I've recompiled it with an up to date version of Turbo compiler, removed MODE calls, removed hard coded referen-

ces to the system variables and linked all extensions required to the program, making it self contained. It now has a configuration block and a new command to allow its display to be moved around high resolution screens. While still a trial release, it does seem to work more reliably on SMSQ/E systems and under pointer environment.

Basic Reporter is freeware nowadays and can be downloaded from:

www.dilwyn.uk6.net/program/index.html"

In case any reader should think we are too fulsome in our praise of Dilwyn, let us just add that he has been a very naughty boy. These news items arrived after our deadline date. Tut! Tut!, Dilwyn. Next time there will be no excuses. Just use your own alarm!

Which Browser?

There was recently some discussion on the QL-users group over the internet browsers QL-ers use. Just Words! had published details of the browsers used to access their website during 2005.

Internet Explorer	72.3%
Netscape	24.5%
Opera	2.2%
Others	1.1%

In this discussion it became clear that many Opera users had the browser configured to identify itself as Internet Explorer and this is, in fact, the default identification. It also became clear that few Opera users were using the latest version of the browser.

QL-ers are obviously fast learners because the equivalent figures for September 2006 were:

Internet Explorer	40.0%
Netscape	30.0%
Opera	30.0%

Over 90% of Opera users are now using the latest version 9.

Ephemeral Website

In his Gee Graphics article in the last issue Herb Schaaf reported that Hirofumi Fukiwara's website was no longer available following the death of the author.

His interest in this site was in a 10 part tutorial on writing sudoku puzzles by hand. Fortunately this tutorial is available elsewhere. The first part can be found at:

http://www.pro.jp/~fuji/numplace/makeproblem/numplace01.html.en

JUST WORDS!

Bigger and Better

Just Words! has upgraded its website. It is is now bigger, better and has more download possibilties:

http://members.lycos.co.uk/geoffwicks/justwords.htm

This is just the start of major changes designed to reflect current developments in the QL Community.

Geoff Wicks, 5b Wordsworth Avenue, Sinfin, Derby, DE24 9HQ, U.K.

Tel: +44 (0)1332 - 271366 *email: gwicks@beeb.net*

Web: http://members.lycos.co.uk/geoffwicks/justwords.htm

North American QL Show - September 27, 2006

by Tony Firshman

Power is a problem

Bill Cable organised the show this year, as Al Boehm is still recovering from his stroke last year. It was in Niagara Falls on the Canadian side. I had visited Niagara two years ago when I stayed with Bill. We were on the way back from a motorcycle swapmeet in Ohio.
This time I again found an American Airlines site that sold tickets in US dollars and again it was 30% cheaper than similar flights from UK sites. The only problem is payment has to be made with an American card (I used Bill's card!).
One time I went to the USA I took my bike - but not planned. I was so delayed that I investigated parking near Heathrow and found a free parking slot within easy cycling but no good by public transport - there was none! This time I parked there intentionally, as I needed the bicycle at the other end (see later). I had a very early morning flight and there was no queue. The reason became clear on the plane - there were maybe 40 passengers in a Boeing 777! At first we were all crammed together in the usual fashion. I was next to a 26 year old US Airforce Iraq conscript who was also a cop in Pittsburgh. On take off he was gripping the seat and sweating "I am scared of flying" - this was somehow re-assuring. It seems US forces are human.
I planned to watch a DVD. I turned on the computer, and it immediately shut down with dead battery light. Odd as I had left it on charge at home. The second battery was OK, but Windows stopped during "Loading personal settings" and reported it could not use my user settings file - a new fault to add to the long Windows list. It said it was booting using Windows defaults. OK - no problem. I would just climb the mountain of re-setting one does when installing Windows. Not so, as the next window reported that any changes made would not be saved on log-off. Hrmm - at last I have found a major flaw in XP.
So:

- Log on as Tony with defaults
- Make a new user
- Log off
- Log on as new user
- Delete user 'tony'
- Make new user 'tony'
- Log off
- Log on as 'tony'
- Delete new user.

Phew. ... but then I thought 's**t' - I have deleted all 'tony' files. Windows is a little clever, and made the new user 'tony.COMPAQ' - but visible name was 'tony'. All my files were still there, so DVD watching became a Windows restore. I suppose I shouldn't complain. This was only the second time in two years, compared with the bi-monthly exercise on the previous W98. Battery though ran out well before the flight was over so I had to watch a bad film.
I arrived at JFK and met an old friend. Fiona lived next door to me in Ascot, and has moved around the world ever since with her banker husband, He actually worked in the twin towers, but was out of town on that fateful day! We filled in the three hours I had before the American Eagle flight to Boston, eating. I wanted to charge my laptop, but there were no power points. Power was a problem, but I had enough battery to show my house re-building photos - I had just finished renovating a house in Aylesbury. I am doing house building for Bill Cable as a break!
At the gate for the Boston flight I found power points. However an American lady was using both. Power was a problem. "No problem - I will unplug my phone" - so she proceeded to unplug her laptop! I then realised I had forgotten my US plug converter - it was in hold baggage. Power was a problem. "Here you can borrow mine" said an Iranian girl opposite". I plugged my computer in and no sign of life. Power was a problem. It looked like either a fuse or a dead power supply. "You can use my power supply" said this very pretty Iranian. It was exactly the same voltage/ current but wrong coax socket. Power was a problem. I tried her fuse in my plug - and indeed, my supply was dead. Power was still a problem.
... so I had to resort to conversation. It turned out this 24 yr old very bright Iranian girl was finding it difficult to get her first job as she kept on speaking her mind, and people don't like that in interviews. They expect you to play the game.
... so onward to Boston, and the Dartmouth Coach Co bus to Lebanon. It took only 10 mi-

nutes from landing to bus stop, so I caught the early bus, saving two hours. What a civilised bus (and driver). It served 'free' water and Pretzels and showed Annie Hall. Brilliant, as I had seen bits of this film but never all. I am looking forward to reminding my teacher friend in Tring of "Those who cannot do, teach. Those who cannot teach, teach German". She is German and teaches German!

I arrived at Lebanon to find American Eagle had broken my bike pedal power system, and the carrier. Power was still a problem. I managed to get the pedal power working but not the carrier. I had to carry my 70lbs on my back. On the way, Mary Boyle (Bill's wife) passed me. "I have a full car, but I can take your back pack" - I think I almost hugged her to death!

Bill and I went through the list of things to do the next day when he was working. One was that the engine on his compressor kept dying. Power was a problem for him too. I went by motorbike to the metropolis (West Lebanon and White River Junction) to try to find a power supply for my laptop. First port of call was Computer Consultants Inc (CC). I missed it, and passed Datacomm Inc. That looked like somewhere to try later. I went back and found CC - but there was a note saying they had moved to Datacomm! It was a real aladdin's cave of old PC bits but no laptop power supplies. Radio Shack was next port of call, and bingo. Juice Igo (Google for it!) is a brilliant product. It is a higher spec mains supply than the one I was replacing, but it also works off 12 volts. I remembered Radio Shack as THE place to go for electronic parts and tools. They have turned into a computer shop.

For the compressor, I bought a spark plug and carburettor cleaner. I stripped the carburettor, cleaned and re-assembled. I also replaced the petrol, sorry gas. It would not start. When Bill got home, he said "Sorry - I forgot to tell you you have to do THIS to remove the load before starting". It ran first time and worked perfectly.

Power was at last not a problem.

I also re-visited the Four Aces diner in White River Junction. Bill Bryson had described it on UK TV years ago as the "second best diner in the US, but I don't like Diners". On the wall there is a review extract. In that he repeats how he doesn't like diners, and says "The Fours Aces food is excellent, unfortunately".

The night before we were due to leave for the show in Niagara Falls, Mary got stuck on the way home with a flat on their relatively new Toyota Prius hybrid car. We were due to leave for the show early Friday morning, so Bill and I arrived at a tyre repair shop very early morning. They could find no fault but suspected a faulty valve which was replaced. We set off for Niagara and arrived about 18:30 uneventfully. There was a good quorum of QLers there, and we tried to organise supper. There was no concensus (of course) so Bill and I went roving looking for suitable restaurants. We ended up at 8pm in the Korean restaurant dead opposite the hotel!

It was very good seeing Al Boehm there in good spirits. He sung grace at the meal.

On our return, we discovered the hotel wifi and proceeded to get our internet kicks for the evening.

In the morning we found very little left of the hotel breakfast so went off to the pancake house opposite!

The show room was excellent, despite half the power sockets not working. There was a good quota of traders ***and*** punters for a change. I am not too sure what actually happened as I spent ***all*** my time on repairs of various types. I managed to get most of the scheduled work done, but didn't manage to get both Jon Kaczor's disk drives or Doug Laverne's monitor working. Bill kindly provided US QLs for spare parts.

Pizzas were provided by Bill for lunch, courtesy of Nesqlug.

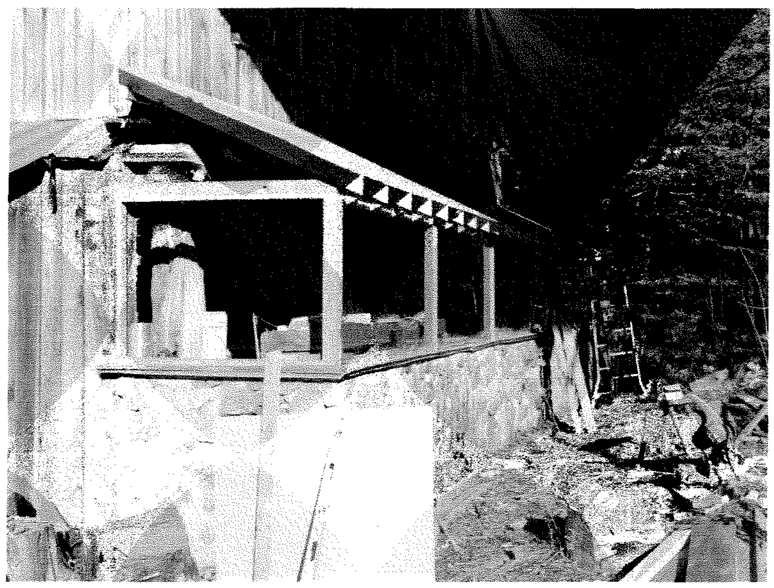
Jim Hunkins spoke on QDT, and Doug Laverne on QL internet connectivity, or the lack of it!

Bill, Mary and I left early as we were driving back overnight. We ate in the "Falls View Restaurant" and had a pretty average meal but an extraordinary view of the Horseshoe Falls. I discovered later, from the ***only*** technical info in a thick glossy tourist guide, that 750,000 US gallons of water flow every second - a quite staggering statistic.

We arrived back at about 3:30am. Up again at 07:30 for rehearsal at church at 08:30! Mary deserves special thanks as I know she only went for me that Sunday! I had been wanting to sing in

Mary's church choir in Meriden for years but had never managed it before. It was good to find that all the music was familiar - Vaughan Williams and Stanford. Even the hymn tunes were standard UK, but not the words! It turned out the priest was in Mary's car on my arrival, so my bicycle and I got a special mention.

The week was mainly spent re-building the side of Bill's "shop" (workshop). This was a job he had been wanting to do for years, but never found the time. It was good to be involved in a more major project than during my previous visits. It was good to see that the running spring water system we installed a few years ago to give tapped water in their kitchen was still operational.



I returned to New York on the following Sunday. Bill was visiting a motorcycle swapmeet in Boston so he gave me a lift. He compiles CD records of the vintage bikes at shows throughout the USA. I was dropped off at N Station in Boston. However we noticed this was on the bike route that Bill had found on the web. This proved a brilliant way into the airport via some very back streets of Boston. Again I saw that the non-tourist areas of Boston are very very run down. It was clear at one point why the route took a very roundabout way. One bridge was marked "Not for bikes" - it was a rusty green 2-decker steel freeway 200 feet above the ground! The usable bridge was an opening one, and the pavement ('road' in the US) made using grids of steel bars. These were fine for cars but not bikes. My small Brompton wheels fell into the gaps!

A bit nearer the airport I asked a local directions, as the map was not too clear at that point. "I am cycling into the airport." "I have lived here for 30 years, buddy, and you cannot cycle into the airport". I carry on, and find the right road. I negotiated a street blocked to cars, went through a mansion block forecourt, and ended up near the Hilton hotel. At that point the map routes a long way around the hotel. On my right I see a road marked "Not for unauthorised vehicles" and taxis and the shuttle buses were driving down it. Now I was certainly not a vehicle and I could see terminal A at the other end, so I cycled easily down there to terminal B.

I was about 8 hours early for my flight, but I was put on standby, and got on a flight 30 minutes later. Worldnews.com had agreed to buy me a return ticket to the UK, so I planned to break my journey and stay in NY with Fiona and family. Fortunately I told the check-in, who were planning to send my bags direct to the UK!

The journey to Larchmont was fine other than an encounter with a very angry train conductor. I was trying to ask people if this was the right train - but they all spoke only Spanish! The conductor was having a long argument with a passenger and was furious at being interrupted, but I could see the train was about to leave! Once the train had started, he told me to remove my bike at the next station. I knew from previous visits that bikes were allowed outside rush hours (and this was Sunday), but it seemed politic simply to fold it into its bag. No more was said by him!

If you ever visit Manhatten, the Bodies exhibition at South Street is mandatory. These are *real* bodies preserved with silicon, and it was quite incredible. Amongst other things, I had not realised how fibre-like nerves actually are. Also seeing the number of smoke-blackened lungs there were was very salutary. .... and I got a senior discount with my driving license!

One interesting feature of this trip was using the SIP phone my son Ben had configured for me. We have an Asterisk server at home, and this handles ***all*** my phones - BT, VOIP and the SIP phone on the laptop. Whenever my laptop is connected to the internet, then my home phone is there for incoming and outgoing calls. I even get answer machine messages as emails. This worked even on Bill's 20k dial-up connection.

I arrived back in the UK on 14 October after another memorable US trip. I hope to be back soon for the next US show.

Colour Themes

by Dilwyn Jones

When Marcel Kilgus gave us Window Manager 2, one of the new features which became available to us was Colour Themes.

Basically, it means you can set up a list of standard colours to be used for window colours, button colours, loose item colours and so on via something called the System Palette, which is a sort of list of colours to be used for the usual pointer environment loose items and so on.

Many modern programs will make use of this facility - they will use this standard list of colours to make sure they have a consistent appearance, so if you set up a colour list with some nice cool grey paper colours, programs like QPAC2, Menu Extension (QMenu) menus, QSpread 4, QD and Launchpad will use the same set of colours - in other words, the same Colour Theme. Until you experience it, you don't appreciate how nice it is to have programs with consistent appearance.

This article sets out to show how to make use of Colour Themes without having to know anything much about the programming theory. Once you have the Theme files, all you have to do is LRUN or EX a short BASIC program to apply the colour theme via the System Palette facility and that's all there is to it!

The term Colour Themes came into being when Wolfgang Uhlig took it upon himself to create a program called QCoCo (reviewed in QL Today Vol. 10 Issue 1) which gave us a user friendly way of creating these Colour Themes without having to mess about with programming at all. QCoCo just saves the list of colours in one file. Wolfgang introduced the name Colour Themes and it's stuck. Much more friendly than more technical sounding terms like System Palette!

Wolfgang Uhlig and I have created several of these colour theme files and they are available from our websites:

My QL site: www.dilwyn.uk6.net/gd2/index.html

Wolfgang Uhlig's QL website:

http://home.planet.nl/~uhlig001/ql/qcoco.html

Applying the Themes

Once you have the Colour Theme files (they are files having names ending with _thm), you need to know how to use them.

The more complex way to use them is to apply them via QCoCo or Launchpad.

The easy way is to run the little basic program in figure 1. This asks for the filename of the theme file and applies it to System Palette 0, which is the one most programs will use by default. Don't worry if you have older programs which don't understand Colour Themes and System Palettes - they will continue to use the old fixed colours like they always did.

Alternatively, if you are an SBASIC user (SBASIC is the version of SuperBASIC supplied with SMSQ/E) you can EX the slightly modified version of the program in figure 2, which will quietly apply the colour theme without even announcing its presence.

```
EX FLP1_THEMES_EX_bas; 'FLP1_BLACK_THM'
```

Figure 1 - Themes_Bas

```
100 REMark apply colour theme to System Palette 0
110 INPUT #0,'Colour theme filename > ';theme$
120 OPEN_IN #3,theme$
130 entries = 57 : REMark number of word length entries in list
140 memory = ALCHP(2*entries)
150 addr = memory
160 FOR addr = addr TO addr+2*(entries-1) STEP 2
170  INPUT #3,colour_word$
180  colour = HEX(colour_word$(2 TO LEN(colour_word$)))
190  POKE_W addr,colour
200 END FOR addr
210 SP_SET memory,0,entries : REMark apply
220 RECHP memory : REMark release heap memory used
230 CLOSE #3
240 PRINT #0,theme$;' applied.'
```

Figure 2 - Themes_Ex_Bas

```
100 REMark apply colour theme to System Palette 0
110 theme$ = CMD$
120 OPEN_IN #3,theme$
130 entries = 57 : REMark number of word length entries in list
140 memory = ALCHP(2*entries)
150 addr = memory
160 FOR addr = addr TO addr+2*(entries-1) STEP 2
170  INPUT #3,colour_word$
180  colour = HEX(colour_word$(2 TO LEN(colour_word$)))
190  POKE_W addr,colour
200 END FOR addr
210 SP_SET memory,0,entries : REMark apply
220 RECHP memory : REMark release heap memory used
230 CLOSE #3
240 QUIT
```

Colour Themes

The files themselves just consist of 57 word-length entries stored as hexadecimal values preceded with a '$' symbol. You don't need to know much about this, anyone wishing to know more is referred to Window Manager documentation and articles such as Wolfgang Lenerz's "New Functionalities in SMSQ/E" in vol. 7 issue 6 and vol. 8 issue 1 of QL Today. Lines which have the most significant bit set (bit 15=1) are 15 bit RGB values, read Window Manager documentation for the formats of other colour values such as those for border types. Once opened, the theme files are poked into a 114 byte table (2 bytes for each of the 57 entries), and a SP_SET command is issued to set the default System Palette 0 to the list of colours in the table, starting from element 0 with 57 entries. Current Window Managers use 57 entries - this may change in future versions if Marcel Kilgus so decides.

QDOS and SMSQ/E

If you have a recent version of SMSQ/E (version 3.00 or later) with Window Manager 2, or QDOS with pointer environment version 2 or later, you should be able to use Colour Themes straight away. SMSQ/E updates can be obtained from your nearest SMSQ/E reseller, while version 2 of pointer environment is available from Wolfgang Lenerz's website:

http://www.scp-paulet-lenerz.com/smsqe/Add1.html

or from the Pointer Environment page on my website:

http://www.dilwyn.uk6.net/pe/index.html

Obviously, on systems which don't have 256 colour or 16 bit colour displays, the system is limited to selections of the four basic colours black, white, red and green. You can still have fun devising simple 4 colour themes for each day of the week if you wish, the point being that you can still enjoy consistency of appearance with recent software. My collections of themes include a few for 4 colour mode 0 use - these have "m4" in the filename.

Where to get suitable Programs

Many programs from Jochen Merz Software and QBranch use the System Palette - QPAC1, QPAC2, QMenu, QD and QSpread to name a few. Suqcess2 database from Bob Spelten (the more recent development of Wolfgang Uhlig's original Suqcess) can also use colour themes.

Recent programs from Per Witte - Msprv, Qwirc, D-Miner and PED for example - are capable.

Most of my recent software is suitable - Launchpad, Q-Trans, Calculator, Calendar, Screen Snatcher, Pic Viewer, Text File Viewer, Reformat, QStarter, QFiles, Sides, Sort & Column Print, QXL.WIN Manager etc are all included and I do hope that all my new QL software will be suitable. I hope over time to produce updated versions of older programs.

Marcel Kilgus has made some programs available on his website to demonstrate use of the new colours, and has 'hacked' or updated versions of some older programs (e.g. Qascade).

Some programs are supplied to make use of the GD2 colours, but do not use the system palette mechanism. Examples are the S-Edit text editor from Ralf Reköndt, current versions of QDT from James Hunkins and programs which use a fixed house-style set of GD2 colours, like those from the Just Words stable. It is important to note that programs which do not recognise colour themes

JOCHEN MERZ SOFTWARE

Kaiser-Wilh.-Str. 302 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://smsq.j-m-s.com smsq@j-m-s.com

THE JMS XMAS-OFFER

ALL PRODUCTS ON THIS PAGE WILL BE SHIPPED WITHOUT POSTAGE CHARGE AND YOU GET AN ADDITIONAL 10 PERCENT DISCOUNT ON THE PRODUCT PRICE. THIS IS TO SAY I'M SORRY FOR NOT BEING AT SEVERAL QL SHOWS AS INITIALLY PLANNED.
OFFER EXCLUSIVELY TO QL TODAY READERS.
VALID UNTIL 24TH OF DECEMBER 2006.
PLEASE STATE "QLTD-XMAS-010" IN YOUR ORDER TO GET THE 10 PERCENT AND THE FREE POSTAGE!
WE WISH YOU A MERRY XMAS AND ALL THE BEST FOR 2007 - AND HOPE TO SEE YOU NEXT YEAR!

SMSQ/E

QPC2 Version 3 with SMSQ/E ... now only **EUR 59,90**
QPC2 Version 3 upgrade from Version 2 ... only **EUR 19,90**
QPC2 Version 3 upgrade from Version 1 ... only **EUR 39,90**
QPCPrint ... still only **EUR 39,90**

Please note that QPCPrint will be shipped on CD only.
QPC2 will also be shipped on CD by default now, as more and more systems (especially notebooks) do not have floppy disks drives built in anymore. If you prefer QPC2 on HD floppy disks, please state with your order. Free updates are available on Marcel's Website www.KILGUS.net If you prefer updates on CD or floppy disk, send in your master medium together with 4 international reply coupons to cover return postage, medium and packaging.

SMSQ/E for ATARI or GoldCard or SuperGoldCard or QXL ...
now only **EUR 39,90**

Colour drivers now included as standard for (Super)GoldCard versions
For shipping inside Europe, EUR 3,- will be added for postage and packaging.
For shipping outside Europe, EUR 6,- will be added for p & p.

If you wish to place your order via internet, please do not send your credit card details in an email!
Please use the secure contact form on the SMSQ homepage: SMSQ.J-M-S.COM
We accept VISA, MasterCard and Diners Club!

will continue to work OK on GD2 systems if you apply a colour theme. They'll just ignore the applied colour theme and carry on working just fine in their old colours like they always did.

Cancelling a Colour Theme

If you wish to reset the system colours after experimenting with colour themes, just use an SP_RESET command from basic.

Conclusion

I hope that this article has shown you that you don't have to be a programmer to make use of Colour Themes, and that you don't have to worry about older programs failing to work if you start using the new colour themes, as older programs just ignore the colour theme.
An interesting but harmless side effect you may run into which just looks a bit odd without causing any errors is a program which uses the Menu Extension. The QMenu menus like File Select and List Select will pick up on the new colour theme while the main program won't. Example of this include Norman Dunbar's Printer-Master 2, which runs in the old-style four colours, or programs from me like Deskjet-A5 and Side-writer which use QMenu menus a lot. When these call up a QMenu menu to do certain functions, these appear in the new colours! Entirely harmless, but does look a little odd at first until you get used to it.
Just using pre-defined themes like the ones mentioned here is enough in itself to make your QLing more colourful! The fact that the major QL software used by most people (QPAC2, QMenu and QD for example) is a good starting point and as more and more authors write new software making use of the new Window Manager facilities, and updated programming tools like Easyptr 4 make it so easy to produce new software, things (and the programs!) are looking good.

Indexing Documents

by David Denham

For a change, I thought I'd send in a short and useful listing. It's a program for indexing text files, which is written in superbasic and easy to modify, tweak and tinker with. Even though the QL has been around for over twenty years, I still find it so easy to write programs like this that I don't think I could ever face up to owning a computer without superbasic on it - a complete indexing program you can easily modify yourself in less than 60 lines of code!
A list of words to be indexed is placed in DATA statements at the end of the program, and it uses this list to scan the document and generate an index text file, listing page numbers at which these words are found.
The text file should be a plain text file with the required number of lines per page as in the printed version of the document - generate such a file with the print to file commands of most word processors and text editors.
It is best if this text file contains no formfeed characters to upset the line count and page number calculations, but if formfeed characters are found, the program tries to compensate by adjusting the line number count to the nearest multiple of the number of lines per page (i.e. where it thinks the next page should start). Not perfect, but adequate for a simple program like this.

Indexer works by making a separate pass through the file for every word to be indexed. This is slow and inefficient, especially for large text files, but it made it much easier to write this program and keep it short enough for publication. A better way of coding this might be to set up a string array with the list of words found and add the page numbers to the array entries to build up the index, then write the array out to disk afterward - this would only need one pass through the text file, but make the program listing longer.

Index Format

The program generates an index as follows:

```
Screen 1,2,4,5
File 1,2,3,4
LRESPR 1
BASIC 1,2,4,6
Editor 1
QL 1,2,3,4,5
Extension 1,2,3
Smsq 1,3
Qdos 1,3
```

The index simply consists of the word searched for, followed by a space, and a list of page numbers in which this word was found, separated by commas. It's quite easy to alter the format by adjusting the PRINT#3 statements in lines 420 to 440 and 540.

The index is a plain text file - this can be loaded into your favourite editor once generated.

Search Words

The words to be indexed are simply entered into the DATA statements at the end of the program. The listing just shows a few QL terms I used to test it, just delete these and add the required search words in their place in lines 660 onward. Indexer uses the EOF function in line 310 to detect the end of the list of search words. The use of INSTR ensures upper and lower case words are matched, so you can enter the words in the DATA statements in the case in which they should appear in the index.

Line by Line Commentary

120-140 - get details required from user (filenames and number of lines per page).

160 - set up the index output file.

210 - check if all index words have been read from DATA.

220 - get the next word.

240 - open the text file to search for current word.

260-280 - variables used in indexing.

310-320 - get a line of text from the file.

330 - increment line number count.

350 - check if word exists on this line.

370 - calculate page number.

380 - prevents duplicate entries in the index from the current page by keeping a record of which page the word was last found on (last_page_no) and the current page.

400-440 - control formatting of index by ensuring that if this is the first occurrence found, the word itself and any spacing before the list of page numbers is sent to the index, followed by the first page number. If this is second or later occurrence, a comma is sent before the next page number to separate the page numbers. Line 450 sets the variable last_page_no to remember the last page number on which this word was found, to prevent duplicate page number entries.

500 - compensate for formfeeds by adjusting line number count to the line where start of next page is expected.

540 - output newline between each word listed in index.

550 - show progress on the screen (word being searched for and number of index entries).

560 - close text file.

590 - close index file.

620 - if you want to see the index on screen after the program has finished, a quick option is to add a VIEW op$ command on this line.

660 - words to be indexed listed in DATA statements from here on.

Suggestions for Tinkering!

To keep readers out of mischief until the next QL Today comes along, here are some suggestions for improving the program.

1. Play around with the output format in lines 420 to 440 and 540. At the moment, the output is the search word, a couple of spaces, and a list of page numbers. The page number list is ragged edged - separate the words and page number lists into columns by tabulating across to a suitable position before printing the page numbers.

2. The program reads the text file once for every word, so that it can output results on one line for each word. Build up the index in a string array in memory instead of writing immediately to disk, this should allow you to write the entire list in one go with only one pass through the text file. Hint: Copy the words from DATA into a wide string array and keep a record of number of finds for each entry, and only write out the array entries where finds > 0.

3. Improve the formfeed handling - at the moment if the formfeed occurs at the start of a line, the rest of the line is treated as being on the previous page irrespective of whether the word is found before or after the formfeed in the line.

4. Superbasic makes it easy for us to edit the DATA statements to alter the search word list. You may find this acceptable, or you may like to alter the program to get the list of search words from a separate text file.

5. The program uses INPUT to get each line of text from a file. On some ROM versions this will fail if lines are longer than 128 characters, or if the last line in the file does not end with a linefeed character. You might like to try error trapping the INPUT# statement for an end-of-file error, remember where the INPUT# statement was in the file (file_pos=FPOS(#4) before the input then go

back here if an end-of-file error has been trapped) and read back character by character to the end of the file.

6. If you are a pointer environment user, use file selection menus such as QMenu to select filenames.

7. Add error trapping to the file opening commands.

8. Short search words may be found within longer words as it stands, for example, the words ON or IN may be found inside words like CONTINUE. To some extent, you can work around this by adding spaces before or after the search word, but you may like to experiment with line 350 to check what is either side of the search word where it was found - think of a list of delimiters for words such as spaces, punctuation marks, numbers, control codes, starts or ends of lines and check either side of the word. A simple example might be if the search word occurs at the start or end of a line of text, and there are no letters immediately to the left or right of it, assume the "match" has been found inside another word.

Have fun!

```
100 REMark QL Document Indexer by David Denham.
110 CLS : CLS #0
120 INPUT #0,'Document filename   > ';ip$
130 INPUT #0,'Index file filename > ';op$
140 INPUT #0,'Lines per page      > ';lpp
150 :
160 OPEN_NEW #3,op$ : REMark The output index file.
170 :
180 PRINT 'Indexing : '; : RESTORE
190 :
200 REPeat indexing_words
210   IF EOF = 1 THEN EXIT indexing_words
220   READ sch_word$ : PRINT ! sch_word$; : REMark Show search word
230   :
240   OPEN_IN #4,ip$ : REMark The plain text file
250   :
260   line_no = 0       : REMark line number in file being read
270   found = 0         : REMark occurrences found of this word
280   last_page_no = -1 : REMark not yet found
290   :
300   REPeat read_lines_of_text
310     IF EOF(#4) = 1 THEN EXIT read_lines_of_text
320     INPUT #4,line_of_text$ : REMark Get a line from the text file.
330     line_no = line_no + 1
340     :
350     IF sch_word$ INSTR line_of_text$ THEN
360       REMark Word has been found in line - Calculate page number.
370       page_no = 1 + INT((line_no-1)/lpp)
380       IF page_no <> last_page_no THEN
390         REMark Only add one occurrence per page to the index.
400         found = found + 1
410         REMark Edit following lines to alter format of the index
420         IF found = 1 THEN PRINT #3,sch_word$;'  ';
430         IF found > 1 THEN PRINT #3,',';  : REMark separator
440         PRINT #3,page_no;
450         last_page_no = page_no : REMark prevent duplicate page entries
460       END IF
470     END IF
480     :
490     REMark if line contains formfeed, adjust line count to next page
500     IF CHR$(12) INSTR line_of_text$ THEN line_no =
(lpp*INT((line_no+lpp-1)/lpp))+1
510   END REPeat read_lines_of_text
520   :
530   REMark if any occurrences of this word found, need a newline
```

```
540   IF found > 0 THEN PRINT #3,
550   PRINT ! "("&found&")" ! : REMark show occurrences on screen
560   CLOSE #4
570 END REPeat indexing_words
580 :
590 CLOSE #3
600 :
610 PRINT #0,'Indexing completed.'
620 REMark Add VIEW op$ command here to view index file created.
630 STOP
640 :
650 REMark list of words to be indexed follows
660 DATA 'Screen','File','LRESPR','BASIC','Editor','QL'
670 DATA 'Extension','Smsq','Qdos'
```

Poxology

by Per Witte

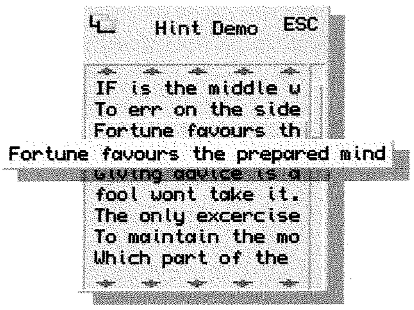
One of the limitations of the PE is that any window opened by a program has to fit inside the main application window (or the Outline, in PE parlance). This can be a bloody nuisance, as you may be aware.

Pox sets out to demonstrate a workaround to this problem. But Pox is more than that. It brings together a number of useful techniques, many of them only recently available, to bring them to the attention of programmers and tinkerers who may not yet be aware of them. To name a few: Pox is a workable Hinting routine, it also demonstrates the use of larger-than-Outline windows and dialogs in programs. It demonstrates some rudimentary inter-job communication. It uses the Home Directory Thing, and shows some simple ways of using the System Palettes. Hopefully both beginners and more advanced programmers may find something of use.

Technical criticism and debate is welcomed.

The examples here use the latest version of **EasyPtr** and **QLiberator** to achieve this, but the principles should be relatively easy to apply to other systems by those who are familiar with them. Though the programs are written for SMSQ/E V3.11+, it shouldnt be too difficult to alter them to work with Qdos/Minerva running PE2.

Pox – short for POpup eXtended – displays a string, that may be wider than the Outline, in a hint window allowing the full string to be shown. (See illustration)



```
IF is the middle w
To err on the side
Fortune favours th
Fortune favours the prepared mind
Giving advice is a
fool wont take it.
The only excercise
To maintain the mo
Which part of the
```

Pox consists of a number of parts. Most of those parts are re-usable as templates for your own programs. You will find the complete package on the current QL Today Cover Disk (which came with this issue) or create your own from scratch following the detailed instructions below.

Either way it is useful to keep all the bits together in a single project folder or directory. For simplicity, I'll refer to this folder as `win1_prg_pox_`. Wherever you see this location referred to, replace it with your chosen location.

Hint Menu

The first thing you need is the Hint menu, like the one used in a previous demonstration of hinting published here in V10/I3/p21. If you don't have it, then briefly, here are the details once again. Alternatively skip down to **Demo Menu**:

Create a null sprite, eg a sprite of size 0, 0 (PE2) or 2,0 the rest. Now fire up Easy Menu and load the saved null sprite into the program. Set that sprite to be the pointer sprite for the present menu definition.

Next choose the outline Attributes and set the border to **Hint Border**. No need to fiddle with any other attributes. Finally DO the Change menu to bring up the outline window metrics and enter the following data into the table:

No.		X		Y		X0		Y0	Object
1	4	24	4	12	0	250	0	92	-> spr

The significant columns here are columns 2 to 5. The rest can be left alone. The '4' in columns 2 and 4 signify that the subsequent dimension can be varied freely. Finally call your menu 'hint' and save it in the project folder.

Demo Menu

For the Demo menu you could re-use the Hintdemo menu described in V10/I3 as for the Hint menu above, or you could create a minimalist demonstration menu in EasyPtr with the following characteristics: A main window with a Move and an Exit button and room enough to take a no less than 124 x 116 pixel Application window, as per the illustration above.

Demo Program

The Demo program contains all the functional code you will need to get extended line hinting in your own programs. My comments normally describe the code block immediately above, unless otherwise stated:

```
1 rem                           Pox Demo V0.01
2 rem                           (C)pjwitte 2oo6
3 :
4 timep%     = 100: rem Timeout before pop-up
5 timed%     = 300: rem Pop-up display time
6 pal        = 0  : rem User-configurable
                    palette 0..3 [or addr]
7 :
```

timep% and ***timed%*** affect the behaviour of the popup window and could be user-configurable. ***pal*** defines the palette number. It can have the values 0 to 3. It could also be user-configurable.

```
8 randomise date
9 :
10 dim dat$(16, 30), ddat$(16, 18)
11 for i% = 0 to dimn(dat$)
12  for j% = 1 to rnd(2 to dimn(dat$(0)))
13   dat$(i%) = dat$(i%)&chr$(rnd(32 to 191))
14  endfor j%
15  ddat$(i%) = dat$(i%)
16 endfor i%
17 :
```

This is where your application would read or generate some data. Here it is replaced by some dummy code. This produces some random length random strings and puts them into two arrays. The first array, ***dat$***, contains the full length string, the second, ***ddat$***, only as much of the string as can be displayed in the Application window.
Note: The last dimension of ***ddat$*** may need to be altered if you are using an Application window of a smaller size than my demo.

```
18 sp_jobpal -1, pal: rem Set system palette
   to use for this and sub-jobs
19 :
```

Sets the System palette for the program to use. See also the Pox program below.

```
20 ch% = fopen(#0; 'con_')
21 mdraw#ch%; home_dir$ & 'hintdemo_men'
22 mawdraw#ch%; 1, ddat$
```

Opens and draws the main program menu. Note the use of ***home_dir$***: This applies only to SMSQ/E versions 3.11+. The hintdemo_men file must then be located in the directory from which the demo is executed. Non-SMSQ/E users will have to hard-wire the location of the menu and accompanying Pox_obj program.

```
23 :
24 wsa = mwdef(#ch%): cia = peek_l(wsa)+ 48:
   rem Pointer to current item
25 :
```

cia points to the location in the Working Definition that is updated with the Current Item (if any) on return from the mcallt call.

```
26 ev% = 0:
   rem Events
27 it% = -1: lit% = it%:
   rem Current and last items
28 sw% = -1: lsw% = sw%:
   rem Current sub window and last sub window
29 dim pv%(15):
   rem Init pointer record
30 :
```

These variables must be initialised before the loop runs. **ev%** could be used for other things, such as timers. Lines 27 and 28 initialise some program switches.

```
31 rep main
32  item = mcallt(#ch%, ev%, timep%)
33  pval#ch%; pv%
```

If an item is hit, or it timed out, a snapshot of the pointer record is taken in ***pv%***.

```
34  if item = -1280 then
35   rem timed out
```

mcallt returns -1280 on timeout, so this is where the hinting action goes:

```
36   lit% = it%: lsw% = sw%
37   it% = peek_w(cia): sw% = pv%(2)
38   if it% = lit% and sw% = lsw%: next main
39 :
```

Switches are set to ensure that the same item doesn't get called up repeatedly. The significance of the Current item (it%) is different in the main window from the application window, so therefore we need two values to judge what is required; one for each (sub-)window. The window the pointer is in is found in ***pv%(2)*** and now in **sw%**.

```
40   sel on sw%
41   = -1: rem Main window
42    sel on it%
43    = 0: Pox 'Move Window'
44    = 1: Pox 'Quit Program'
45    = remainder: Pox 'Furniture ' & it%
46    endsel
```

Only the standard buttons are found in the main window, such as Move and Exit. Anything else is trapped, for the sake of this demo.

```
47   = 0: rem Sub window 0
48    sel on it%
49    = -250: Pox 'Blank'
50    = 28000 to 32767: Pox 'Dont know ' & it%
51    = remainder: Pox dat$(it%)
52    endsel
53   endsel
```

This handles the Application window, trapping the furniture and any object or action it doesn't know. The relevant line is pointed to by ***it%***, derived directly from the Current Item item number. The line can now be sent to Pox to be displayed in its full glory.

```
54  else
55   if item < 0 then
56    select on item
57    = -1: rem Wmove (but never gets here)
58    = -2: quit
59    endsel
60   endif
61   sw% = -1111: rem Reset
62  endif
63 endrep main
```

Above are the normal action routines of the program. Not much in the case of this demo.

```
64 :
65 def proc Pox(txt$)
66 loc id
67 id = few(home_dir$ & 'Pox_obj'; hex$(pal,
   32) & txt$)
68 enddef Pox
69 :
```

The Pox function merely calls up the program that does all the hard work. For the meaning of the parameters, see the Pox program, below. Note the use of ***home_dir$*** again. ***id*** could return a message from the called job: a negative intger representing an error, or any other return value. This would be returned by the calling job with QUIT ‹n› (with ‹n› being some return code).
The ***pal*** parameter is not used to good effect in this demo. In a real program you'd allow your user to specify any of the four built-in system palettes. If you had a house livery, for example, you could specify your own palette, or choice of palettes. If you were feeling particularly generous, you might even allow the user to devise his own palette specifically for use with your program. Either way, Pox will handle it. For Pox ***pal*** can have a value from 0 to 3 or the address of a valid palette definition in memory.
Beware: Any value outside 0..3 is taken to be an address, so carelessness could crash the machine.

Pox Program

You've seen how the Demo program used the ***home_dir$*** to find its menu definition. I could well have used this technique here, but thought I'd move on to something more advanced. The program is to be compiled. The following steps should be familiar to those who frequently use EasyPtr:
You need to use EasyPtr's Appman utility to prepare the hint menu for appending to the compiled code. Simply start up Appman and specify that you want to load a menu. Load the hint_men menu prepared earlier and save the file as Pox_app. Then load it into S*Basic with LRESPR. Modify line 1 below, according to where you put it. Type in and compile the lines below and place the object file in the pox project folder.

```
1 remark $$asmb=win1_prg_pox_Pox_app,0,10
2 remark $$chan=2
3 remark $$stak=128
4 remark $$heap=512
5 :
```

Show QLib where the binary is, and set the memory overheads. These values are rough estimates.

```
6 rem POX - POpup eXternal - External Hint
7 :
8 rem (C)pjwitte 2oo5
9 rem V0.01 November 17th 2005
10 :
11 sp_hintbd%       = 552: rem Hint border
12 sp_hintbg%       = 553: rem Hint background
13 sp_hintfg%       = 554: rem Hint foreground
14 :
15 if len(CMD$) < 9: quit
16 c$ = CMD$
```

If the command line is wrong, whatever executed this program didn't mean to and the program terminates without a squeak.

```
17 mn_pox = appa0("hint"): rem Menu address
18 :
```

Get the address of the menu definition

```
19 pal = hex(c$(1 to 8))
20 if pal > 3 or pal < 0 then
21  sp_jobownpal -1, pal: rem Use calling
```

```
job's private palette
22 else
23  sp_jobpal -1, pal: rem Use system palette
24 endif
25 :
```

If pal is an address (presumably somewhere in the calling job's heap) use that, otherwise use system palette 0..3 (The standard system palettes all use a yellow hint window background, so you won't see much difference between them. The technique is more useful with other external objects you could implement.)

Warning
Game in progress!
If you continue the game is lost!

```
26 c$ = c$(9 to)
```

The rest of the parameter is the text to display.

```
27 dim pv%(15)
28 :
29 ch% = fopen(#0; "con_")
30 rdpt#ch%; 48: pval#ch%; pv%: close#ch%
31 x% = pv%(14): y% = pv%(15) + 2
```

Find out pointer location, but discard window again as it interferes with operations..

```
32 l% = len(c$) * 6 + 4
33 xs% = x% - (l% div 2): if xs% < 1: xs% = 1
34 :
```

Calculate size of window required. (There is no contingency for text that exceeds the width of the screen!)

```
35 ch% = fopen(#0; "con_")
36 mdraw#ch%; mn_pox, xs%, y%, l%, 14
37 rdpt#ch%; 48, x%, y%
38 wm_paper#ch%; sp_hintbg%: wm_ink#ch%;
sp_hintfg%
39 cursor#ch%; 2, 2: print#ch%; c$;
40 rdpt#ch%; 9
41 :
```

Open and display hint window and show the text. Die on a direct click or the slightest movement of the pointer – even if the window is buried.

The concept of using external dialogs or utilities can of course be widened. QMenu's file selector and other dialogs lend themselves particularly well to such use. No excuse anymore taking up half the screen with two buttons and a line of text because you need the window for File Select later!

Programming in Assembler – Part 16 - Using the Maths Package

by Norman Dunbar

Don't worry, it's not as bad as it sounds. What I'm talking about is the internal package of routines, provided by the operating system, to allow various mathematical operations to be carried out. For example, multiplying two floating point numbers together, or finding the square root of a number and so on.
The two entry points to this useful set of routines is known (in old QDOS format) as RI_EXEC – which carries out a single operation – and RI_EXECB – which carries out a stream of operations. For SMSQE users the names are QA_OP and QA_MOP respectively.
These are vectored routines which simply means that you can find where they are by loading the contents of a word in memory. If the actual location of the code moves around between versions of SMSQE (As QDOS is not being updated) then the vectors remain in the same place.

To call a vectored routine is quite simple. All you do is set up the entry registers as per the QDOSMSQ documentation, load an address register with the vector and JSR (An) as follows :

```
        move.w   ca_gtfp,a2          ; Fetch floating point parameter(s)
        jsr      (a2)                ; Do it
```

On return, D0 will be set to an error code or zero if it all worked ok. All current vectors are WORD sized by the way.
Without any further hesitation, lets jump straight in with some example code. The following short routine shows the RI_EXEC entry point to the maths package in use. It is a simple demonstration and creates a new function names ROOT which simply returns the square root of its single parameter.

```
*-----------------------------------------------------------------------------
* Equates as required.
*-----------------------------------------------------------------------------
err_bp      equ     -15                 ; Bad parameter error
bv_rip      equ     $58                 ; Maths stack pointer
ri_sqrt     equ     $28                 ; Op code for square root

*-----------------------------------------------------------------------------
* Usual start block for PROCedure and FuNction extensions.
*-----------------------------------------------------------------------------
start       lea     define,a1           ; Pointer to the definition table
            move.w  BP_INIT,a2          ; The vector we need to use (= $110)
            jsr     (a2)                ; Call the vectored routine
            rts                         ; And return any errors back to SuperBasic

*-----------------------------------------------------------------------------
* Definition block for our new function.
*-----------------------------------------------------------------------------
define      dc.w    0                   ; 0 new procedures
            dc.w    0                   ; End of procedures

            dc.w    1                   ; There is 1 function

            dc.w    root-*              ; First function
            dc.b    4,'ROOT'

            dc.w    0                   ; End of functions

*-----------------------------------------------------------------------------
* The actual start of the ROOT code is next.
*-----------------------------------------------------------------------------
root        move.l  a5,d7               ; End of parameters
            sub.l   a3,d7               ; Minus start of parameters
            cmpi.w  #8,d7               ; Do we have a single parameter ?
            beq.s   get_1               ; Yes
bad_param   moveq   #err_bp,d0          ; Bad Parameter error
quit        rts                         ; Exit.

*-----------------------------------------------------------------------------
* The single floating point parameter is fetched next.
*-----------------------------------------------------------------------------
get_1       move.w  ca_gtfp,a2          ; We want a floating point variable
            jsr     (a2)                ; Fetch it
            tst.l   d0                  ; Did it work ?
            beq.s   got_ok              ; Yes it did
            rts                         ; Bale out with error code otherwise

*-----------------------------------------------------------------------------
* Check that it all worked.
*-----------------------------------------------------------------------------
got_ok      cmpi.w  #1,d3               ; Make sure we only got one parameter
            bne.s   bad_param           ; Oops !

*-----------------------------------------------------------------------------
* The value on the arithmetic stack is ready to be SQRTed.
*-----------------------------------------------------------------------------
do_it       moveq   #ri_sqrt,d0         ; Take square root
            moveq   #0,d7               ; Must be zero or crash !
            move.w  RI_EXEC,a2          ; Get vector
            jsr     (a2)                ; Do it
            tst.l   d0                  ; Was it ok ?
            bne.s   quit                ; Oops !
```

```
*------------------------------------------------------------------------
* If all went well, return the new value on the arithmetic stack as a float.
*------------------------------------------------------------------------
ret_fp      moveq   #2,d4               ; Return FP number
            rts                         ; Exit with result
```

Save the above code to a file (mine is called square_root_asm) and assemble it. Once done, LRESPR the resulting bin file (square_root_bin) and try it out as follows:

```
PRINT ROOT(9)
PRINT ROOT(100)
PRINT ROOT(25)
```

and so on. You can make sure that it is working properly by comparing the result from ROOT with the corresponding result for SQRT.

There is nothing complicated in the code. Most of the above is checking that we expect a single parameter and checking that everything worked on and so on. It is the last 8 lines of code that do the actual work and return the result to SuperBasic.

The example above shows how a single operation is carried out. What do you have to do if the mathematical operation you want to perform takes more than a single step?

The answer is simple, you build a list of steps as byte values and terminate them with a zero byte, then call RI_EXECB to execute the steps in order.

Here is another example which uses a relatively simple set of commands to work out the Nth root of any number. Sounds complicated but it is quite simply done using about the only bit of maths 'trickery' that I can remember from my time at school.

The following simple SuperBasic code will demonstrate:

```
1000 DEFine FuNction AnyRoot(m, n)
1010 :
1020 REMark Returns the Nth root of the number M
1030 :
1040 LOCal ln_m
1050 :
1060 ln_m = LN(m)
1070 ln_m = ln_m / n
1080 RETurn EXP(ln_m)
1090 END DEFine
```

If you type the above into SuperBasic and call it as follows, you can calculate all the roots you want:

```
PRINT AnyRoot(100, 3) : rem calculate the cube root of 100
```

And so on. The code works and works quite well, however, as this is an Assembly Language tutorial series, I can't let you off the hook that easily! Here's the Assembly version.

```
*------------------------------------------------------------------------
* Equates as required.
*------------------------------------------------------------------------
err_bp      equ     -15                 ; Bad parameter error
bv_rip      equ     $58                 ; Maths stack pointer
ri_ln       equ     $2a                 ; Take LN of a number
ri_div      equ     $10                 ; Divide Top of stack into next on stack
ri_exp      equ     $2e                 ; EXP of a number
ri_end      equ     $00                 ; End of opcodes list

*------------------------------------------------------------------------
* Usual start block for PROCedure and FuNction extensions.
*------------------------------------------------------------------------
start       lea     define,a1           ; Pointer to the definition table
            move.w  BP_INIT,a2          ; The vector we need to use (= $110)
            jsr     (a2)                ; Call the vectored routine
            rts                         ; And return any errors back to SuperBasic
```

QUANTA

Independent QL Users Group

World-wide Membership is by subscription only,
offering the following benefits:
Bimonthly Newsletter – up to 40 pages
Massive Software Library – All Free !
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for UK members
Overseas subscription £17

Barclaycard: Visa: Access: MasterCard: Accepted

Now in our Twenty Third Year

Further details from the Membership Secretary

John Gilpin, 181, Urmston Lane
Stretford, Manchester, M32 9EH (UK).
Tel. +44 (0) 161 865 2872

Or

Visit the Quanta Web Site
http://www.quanta.org.uk
E-mail: membership@quanta.org.uk

```
*------------------------------------------------------------------------
* Definition block for our new function.
*------------------------------------------------------------------------
define      dc.w    0                   ; 0 new procedures
            dc.w    0                   ; End of procedures

            dc.w    1                   ; There is 1 function

            dc.w    anyroot-*           ; First function
            dc.b    7,'ANYROOT'

            dc.w    0                   ; End of functions

*------------------------------------------------------------------------
* The actual start of the ANYROOT code is next.
*------------------------------------------------------------------------
anyroot     move.l  a5,d7               ; End of parameters
            sub.l   a3,d7               ; Minus start of parameters
            cmpi.w  #16,d7              ; Do we have two parameters ?
            beq.s   get_2               ; Yes
bad_param   moveq   #err_bp,d0          ; Bad Parameter error
quit        rts                         ; Exit.

*------------------------------------------------------------------------
* The two floating point parameters are fetched next.
*------------------------------------------------------------------------
get_2       move.w  ca_gtfp,a2          ; We want floating point variables
            jsr     (a2)                ; Fetch
            tst.l   d0                  ; Did it work ?
            beq.s   got_ok              ; Yes it did
            rts                         ; Bale out with error code otherwise

*------------------------------------------------------------------------
* Check that it all worked.
*------------------------------------------------------------------------
got_ok      cmpi.w  #2,d3               ; Make sure we only got one parameter
            bne.s   bad_param           ; Oops !

*------------------------------------------------------------------------
* A list of op codes to calculate the Nth root of M.
*------------------------------------------------------------------------
op_codes    dc.b    ri_div               ; Divide TOS into NOS
            dc.b    ri_exp               ; Take EXP of TOS
            dc.b    ri_end               ; End of op codes

*------------------------------------------------------------------------
* At this point there are two values on the stack :
*
* 0(A6,A1.L) = M = Big value
* 6(A6,A1.l) = N = Root to find
*
* To work out our Nth root of M, we need to do the following :
*
* Take the LN of M.
* Divide it by N.
* Take the EXP of the result.
* Return it to SuperBasic.
*

* Of course, it's never as easy as it seems !
*------------------------------------------------------------------------
do_it       moveq   #ri_ln,d0           ; LN op code
            moveq   #0,d7               ; Must be zero or crash !
            move.w  RI_EXEC,a2          ; Get vector
            jsr     (a2)                ; Do it
            tst.l   d0                  ; Was it ok ?
            bne.s   quit                ; Oops !
```

```
*------------------------------------------------------------------------------
* Now the stack is holding the following :
*
* 0(A6,A1.L) = LN(M)
* 6(A6,A1.L) = N = Root to find.
*
* They are the wrong way around :o(
*------------------------------------------------------------------------------
swap_tos    move.l  0(a6,a1.l),d7       ; Get a long word
            move.l  6(a6,a1.l),d6       ; And another
            exg     d6,d7               ; Swap them around
            move.l  d7,0(a6,a1.l)       ; Store
            move.l  d6,6(a6,a1.l)       ; Store
            move.w  4(a6,a1.l),d7
            move.w  10(a6,a1.l),d6
            exg     d6,d7               ; Swap again
            move.w  d7,4(a6,a1.l)
            move.w  d6,10(a6,a1.l)      ; Now we have N and LN(M) swapped

*------------------------------------------------------------------------------
* The stack is how we want it to be, so we can continue.
*------------------------------------------------------------------------------
do_more     moveq   #0,d7               ; Or a crash will probably result
            move.w  RI_EXECB,a2         ; Perform a stream of ops
            lea     op_codes,a3         ; Op codes to perform
            jsr     (a2)                ; Finish the ops
            tst.l   d0                  ; Was it ok ?
            bne.s   exit                ; Oops !

*------------------------------------------------------------------------------
* If all went well, return the new value on the arithmetic stack as a float.
* Note that the maths stack is 6 bytes shorter now, so we have to save the top
* in BV_RIP before we exit.
*------------------------------------------------------------------------------
ret_fp      move.l  a1,bv_rip(a6)       ; Make sure maths stack is set
            moveq   #2,d4               ; Return FP number
exit        rts                         ; Exit with result
```

Save the above code to a file (mine is called any_root_asm) and assemble it. Once done, LRESPR the resulting bin file (any_root_bin) and try it out as follows:

```
PRINT ANYROOT(9, 2)
PRINT ANYROOT(100, 3)
PRINT ANYROOT(25, 4)
```

There's not much of real interest in the above code. As ever we validate our parameters to make sure we only expect two then fetch them as floating point values onto the maths stack. After a check to see that we really did get two parameters, we have the values M and N on the stack with M being at the 'top' (TOS = top of stack) and N being underneath it (NOS = next on stack).
We start by running a single op code to take the LN of M which leaves the stack with a new TOS which is the LN value for M.
We next want to divide LN(M) by N but unfortunately, they are the wrong way around so we swap over the 6 bytes at 0(A6,A1.L) with the 6 bytes at 6(A6,A1.L) and then run a sequence of op codes to:

Divide LN(M) by N leaving the result as the TOS.
Take the EXP of LN(M)/N as the new TOS

Once this has been done, we store the new value of A1 at BV_RIP(A6) as required, set the result to be a floating point number and exit to SuperBasic with the result.
As you may have noticed, the text above mentions that the math stack pointer (A1.L) can be changed by the various op codes that we execute. The following table gives you details on what op codes are available and how they manipulate the maths stack.

Val	OpCode	A1.L	Description
00	RI_END	=	End of op code list (RI_EXECB)
02	RI_NINT	+4	Convert FP to Word INT
04	RI_INT	+4	Truncate FP to Word INT
06	RI_NLINT	+2	Convert FP to Long INT
08	RI_FLOAT	-4	Convert Word INT to FP
0A	RI_ADD	+6	Add TOS to NOS, remove TOS from stack
0C	RI_SUB	+6	Subtract TOS from NOS, remove TOS from stack
0E	RI_MULT	+6	Multiply NOS by TOS, remove TOS from stack
10	RI_DIV	+6	Divide TOS into NOS, remove TOS from stack
12	RI_ABS	=	Make TOS positive
14	RI_NEG	=	Negate TOS
16	RI_DUP	-6	Copy TOS and create a new TOS above current TOS.
18	RI_COS	=	Cosine of TOS
1A	RI_SIN	=	Sine of TOS
1C	RI_TAN	=	Tangent of TOS
1E	RI_COT	=	Cotangent of TOS
20	RI_ASIN	=	Arcsine of TOS
22	RI_ACOS	=	ArcCosine of TOS
24	RI_ATAN	=	ArcTangent of TOS
26	RI_ACOT	=	ArcCotangent of TOS
28	RI_SQRT	=	Sqare root of TOS
2A	RI_LN	=	Natural log of TOS
2C	RI_LOG10	=	Log base 10 of TOS
2E	RI_EXP	=	Exponential of TOS
30	RI_POWFP	+6	Raise NOT to power TOS, remove TOS from stack
FF31-FFFF Are the 'save' and 'load' op codes. See below for details.			

So, there you have it, a pile of ingredients all set for you to make up your own numerical recipes. Have fun.

Now, one thing that I have not mentioned above, or even used in the code examples is temporary storage. However, before I delve into that, it's best if I show you exactly what input and output registers are required for the RI_EXEC and RI_ECXECB vector calls.

RI_EXEC - Execute a single maths package op code.	
Entry Registers.	
D0.W	Op code. The high word of D0 should be zero.
D7.L	Should be zero.
A1.L	Arithmetic stack pointer (relative to A6)
A4.L	Pointer to variable storage (relative to A6)
Exit Registers.	
D0	Error code.
D1-D3	Preserved
A0	Preserved
A1	Updated to new arithmetic stanck pointer
A2-A4	Preserved.

RI_EXECB - Execute a stream of maths packages op codes.	
Entry Registers.	
D7.L	Should be zero.
A1.L	Arithmetic stack pointer (relative to A6)
A3.L	Pointer to list of op codes. (relative to A6)
A4.L	Pointer to variable storage (relative to A6)

```
| Exit Registers.                                  |
+--------------------------------------------------+
| D0      Error code.                              |
| D1-D3   Preserved                                |
| A0      Preserved                                |
| A1      Updated to new arithmetic stanck pointer |
| A2-A4   Preserved.                               |
+--------------------------------------------------+
```

You will notice that A4 was never used in my two examples. This is a pointer to the top of an area of memory where you wish to save floating point values to, and load them back from. A4 is relative to A6 (as ever).
The op codes from $FF31 (-207) through $FFFF (-1) can be used to save and load 6 byte floating point values from the stack to and from the variables area.
Op codes that are even allow numbers to be loaded from storage onto the stack creating a new TOS and setting A1 to A1-6.
Op codes that are odd cause the number at TOS to be removed from the stack and saved in the variables area. This causes A1 to change to A1+6. The corresponding load routine is the op code minus 1. (If I call this routine with $FF33 then the opposite routine is $FF32 and so on.)
The actual start address of the variables area where your number will be stored is calculated as:

A6.L + A4.L + (D0.W AND $FFFE)

Each load or save operation uses 6 bytes starting at the above address and working UP in memory. This means that you cannot use all of the load/save op codes for the following reason.
Assume you want to save two numbers from the stack. You might be tempted (as I was) to assume that you could save the first using $FFFF and the second using $FFFD. OK, try it out. Remember saves are odd, loads are even.
Assume also that the absolute address (ie A6 + A4) of your variables area is $1000 0000.
So, where do our two values end up at?

$FFFF -> $1000 0000 + ($FFFF AND $FFFE) = $1000 0000 + $FFFE = $0FFF FFFE for 6 bytes.
$FFFD -> $1000 0000 + ($FFFD AND $FFFE) = $1000 0000 + $FFFC = $0FFF FFFC for 6 bytes.

Because each save uses 6 bytes, the ranges covered are:

$FFFF -> $0FFF FFFE to $1000 0003 (oops!)
$FFFD -> $0FFF FFFC to $1000 0001 (oops!)

This has two pretty major problems in my opinion. The first is that we have overwritten some bytes above the top of our variables area and the second is that we have managed to overwrite a few bytes of our first saved number with the second one!
The maximum range of bytes available for saving data to and loading it back from is between -208(A6,A4.L) for op code $FF31 to -2(A6,A4.L) for op code $FFFF, however, it seems that you are best to use only certain values (see below) to avoid trashing your saved values and avoid using the top two values $FFFF and $FFFD for saves and loads or you will partially overwrite other data above your variables area.
I would advise using the save codes as follows:

$FFFB (-5) as the absolute minimum value; then
$FFF5 (-11)
$FFEF (-17)
$FFE9 (-23)
$FFE3 (-29)
$FFDD (-35)
$FFD7 (-41)
...

And so on subtracting 6 from the op code each time. To load these values back onto the arithmetic stack, use the following codes:

$FFFA (-6) as the absolute minimum value; then
$FFF4 (-12)
$FFEE (-18)
$FFE8 (-24)
$FFE2 (-30)
$FFDC (-36)
$FFD6 (-42)
...

WARNING: If you have a copy of Pennell's QDOS Companion book, the documentation for the load and save op codes is wrong. He mentions that they are byte sized, not word, and range from $32 to $FF - this will work (even on SMSQE) but only as an undocumented feature.
(Thanks Marcel). Pennell also gets the address range wrong as -206(A6,A4.L) to -2(A6,A4.L).

WARNING: If you have Dickens' Advanced QL User Guide, he too gets it slightly wrong. His calculation of the actual address where the FP value will be stored is (A6+A4+((D0.W OR $FF00) AND $FFEE)) - it should be AND $FFFE as only bit 0 is cleared.

Ok, that's a quick overview of the maths package that is built into QDOSMSQ. I'm off on holiday to France tomorrow, so I'll see you (as it were) when I get back. Happy coding.

Perfect Partners

by Geoff Wicks

Many, many moons ago, when I was responsible for the Quanta helpline, a member asked about snatching QL screens and converting these to a PC format. As it happens he was a QPC2 user and knew there was the "cheat's method" of pressing CTRL + PrintScreen to save the image to the Windows clipboard, but he was interested to know the "official" way of snatching and converting.
At the time I was using the "cheat's method" and realised I had no idea how to capture and convert GD2 screens in another way. I appealed for help and received only one reaction. This was from David Bunbury and was sufficient to point me in the right direction.
Way back in far more distant past, all you needed to know to save a screen was the simple command:

```
SBYTES flp1_screen_scr,131072,32768
```

Then things became more complicated. QL systems were developed that had screens larger than the standard 512 x 256 pixels. This problem was relatively easily solved by the _pic format, which saved additional information about the screen size and mode.
Simplicity soon disappeared when the further complication of high colour systems came along. Each QL platform had its own implementation of the new colours and so in addition to modes 4 and 8 we now had modes 16, 32 and 33. (In fact I believe there are even more. Didn't the Thor have a mode 12? And were there other modes for Atari?)
When I started my investigation of snatching and converting high resolution high colour screens, I soon discovered the snatching was easy, but the converting was not. The conversion programs all had their limitations, and usually this was because they were only compatible with one platform. Sometimes it was necessary to convert screens between QL formats before you could convert them to a PC. To make things even more difficult not all the programs were bug free, and although corrections had been published in the QL press, there was a huge paper trail you had to follow to find the correct codes and procedures.
The article I had promised Quanta members was never written. I never got beyond the first few stages of research. And, even if I had tried to write the article, it would have been too long and complicated to fit into the pages of the Quanta Magazine. The article remained on my "to do" list until when events overtook me last summer. Two QL programmers have developed software that are perfect partners in snatching and converting screens.

Snatching

Software for snatching QL screens has existed for a long time and most has been kept up to date by the authors to ensure compatibility with high resolution and high colour screens.

There are two ways of saving complete screens. The SBYTES command detailed above and a program from Dilwyn Jones called "Screen Snatcher". The former can only be used with the traditional QL screen and colours, but the latter is much more versatile and can also be used for saving screens from programs not containing a screen saver option.

Screen Snatcher is a program that sits in the background of your machine and saves the screen to ram disk or other media of choice when a certain key combination is pressed. There have been several versions of Screen Snatcher and the latest forms part of Dilwyn's launchpad suite. It can also be downloaded from his web site:

www.dilwyn.uk.net/graphics/index.html

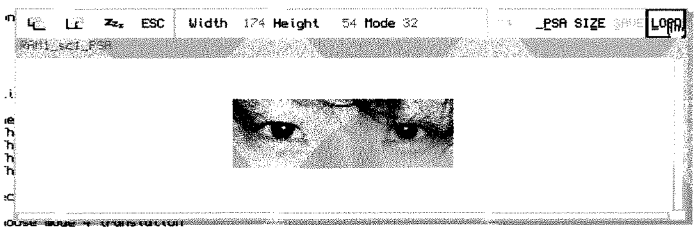
Sometimes you do not want to save the full screen but only a part of it. There are two programs that do this GraBIT, and SVSCR.

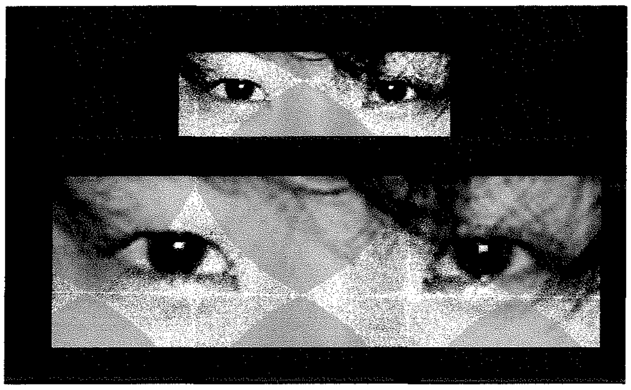
Of these GraBIT, written by Phil Jones, is the older program. It is a pointer program that appears as a button when loaded. If you click on this a "top left hand" cursor appears which you move to the top left hand corner of the part of the screen you wish to save. A HIT transform the cursor to a bottom right hand one and you can move this to the position you want. A DO then saves the partial screen in _pic format to your media and file name of choice.

GraBIT is available on disk UG18 in the Quanta library and can also be downloaded from Dilwyn's graphics page.

SVSCR is a later partial screen snatcher suite from George Gwilt that I reviewed in QL Today volume 10 issue 4 page 49. Since I wrote the review George has done a lot of work to improve the compatibility, versatility and user-friendliness of the suite and that includes adding _pic support. I have no hesitation in naming the SVSCR suite as one of the perfect partners in snatching and converting.

When I reviewed the SVSCR suite I was unable to illustrate the resizing of saved partial screens as this did not work on all QL systems including QPC2. George has now written a program, PMOV, which is compatible with most QL systems, and which supports modes 4, 16, 32 and 33. I do not intend to go into any detail of how to use this program, as this is fairly straightforward. I have included two screen shots to illustrate it. The original picture was a jpg image of a face loaded into the QL using photon. I then saved the eyes as a partial screen which the first illustration shows loaded into PMOV before resizing. The second illustration shows the original partial screen together with the resized version.





Be warned! In this issue of QL Today not only is Big Brother watching you, but also Bigger Brother!

George Gwilt SVSCR's suite can be downloaded from the SQLUG web site:

www.jms1.supanet.com

Finally to end this section on snatching screens, there is a short table showing what is possible with each of the four ways of doing this.

	PE Required	QL Screens	HiRes screens	Partial screens	QL Colours	GD2 colours
SBYTES	×	✓	×	×	✓	×
Scr Snatch	× ✓	✓	✓	×	✓	✓
GraBIT	✓	✓	✓	✓	✓	✓
SVSCR	✓	✓	✓	✓	✓	✓

Converting

One of the best programs for converting mode 4 and mode 8 screens is Jonathan Hudson's QUNPIC. This converts _pic files into bmp, png, postscript, TIFF or pcx formats. I mention this program in passing because it can handle some files that are not suitable for conversion via the other program I am about to describe. I shall return to this point later.

QUNPIC can be downloaded from:

www.daria.co.uk

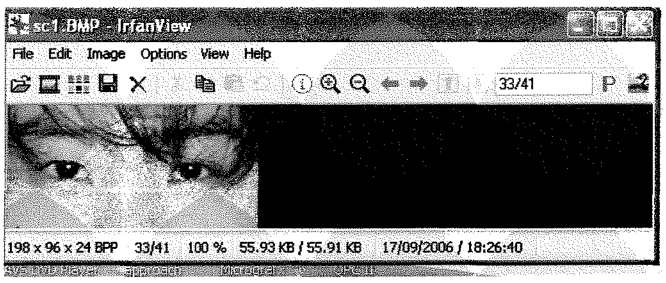
During the summer Dilwyn Jones released an alpha version of the nearest thing the QL has to a universal conversion program. This has now been upgraded to a beta version, and, although this means the final version is yet to come, I have no hesitation in naming it the second of my perfect partner programs.

Dilwyn's program, BMP_obj, will convert practically any file in any QL mode to the PC's BMP format. This article is not intended to be a full review of the program - QL Today would welcome this from anyone who makes extensive use of it - but a quick description.

Our illustration shows the screen you get on loading BMP_obj. You will see that it can handle the QL's _scr, _pic and _psa formats, and that it is not only able to convert from the QL to the PC, but also from the PC to the QL. In addition there is a picture viewer and an option to convert between different QL modes.

You will see from the illustration that the use of the program is fairly straightforward. To convert a screen you click on the QL to BMP button and then just enter source file to be converted and then destination drive.

Just to show that the program does work, the final illustration shows a converted image of our big brother eyes in a PC graphics program.


sc1.BMP - IrfanView
File Edit Image Options View Help
33/41
198 x 96 x 24 BPP 33/41 100 % 55.93 KB / 55.91 KB 17/09/2006 / 18:26:40

There is just one big disadvantage to this program. PC images in BMP format are uncompressed and this means that you will need huge amounts of memory to convert some images. An example of this is the conversion of LineDesign pages. LineDesign has a routine for printing pages to a magnetic or other medium in _pic format. I regularly use this to convert LineDesign pages to a PC graphics format, and up till now have used Jonathan Hudson's program. A LineDesign _pic file is a massive 2376 x 3368 pixels that Dilwyn estimates would generate a BMP image of 24Mb! (There are also problems with the format of LineDesign generated files, but Dilwyn is working on this.)

Incidentally if you are interested in converting LineDesign pages to a PC format, I described how to do this in QL Today volume 5 issue 2 page 20. A shortened version can also be downloaded from the Just Words! help and advice web page:

http://members.lycos.co.uk/geoffwicks/justwords.htm

It is probably over a year since I first promised to write this article. If it had not been for the efforts of George Gwilt and Dilwyn Jones during the summer months it would probably never have been written. For any graphics conversion between the QL and a PC try the perfect partners, SVSCR and BMP.

DOC Viewer

by David Denham

This is a short review of a very simple (and free!) Quill document viewer. Plain text file viewers are ten a penny. This one is different - it lets you view Quill _DOC files without seeing the usual garbage caused by the internal codes of a Quill DOC file.

It's a pointer driven program from Pal Monstad in Norway. It needs the Jochen Merz menu extension installed, and also seems to need a ramdisk (it seems to use ram7_).

It comes in two versions, one with QLIB_RUN built in, the other without QLIB_RUN for those who already have the Q-liberator compiler extensions on their QL.

When you start the program, you get a simple, uncluttered display (see figure 1) with just three commands along the top. These are the Move symbol, a View File command for selecting filenames and an ESC command to quit from the program.

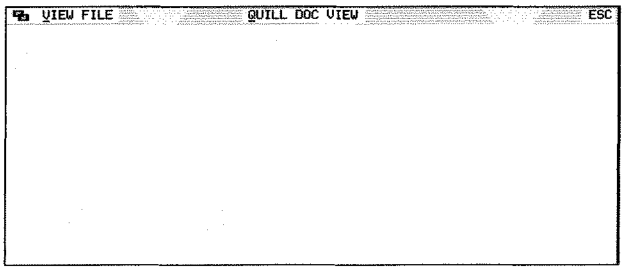
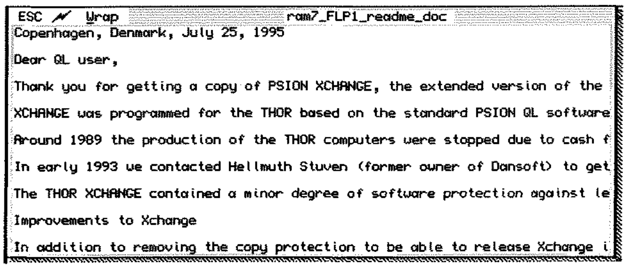


Figure 1

When you select VIEW FILE, you get the usual file select menu making loading and selecting a file very easy and familiar - most of us have used the Jochen Merz file selection menu in some program or other at some point!

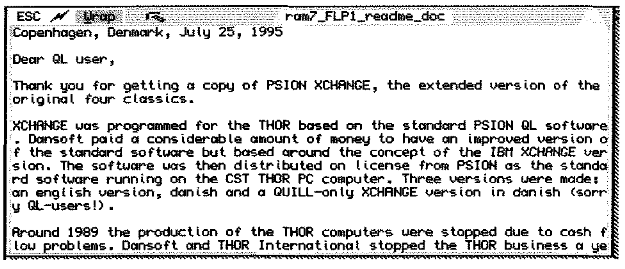
Helpfully, the file selection menu lists just Quill DOC files. In fact, it only lists files having filenames ending with _DOC, although you can change this if you give your DOC files a different suffix. I tried to fool it by feeding it a .DOC file from a PC program, but it survived that by politely telling me that "This is not a Quill Doc file."
Once given the right sort of file to play with, it proceeds to load and display the document. If you have ever tried copying a Quill Doc file to the screen you will have found that either the document is hard to read because lines of text are truncated, or the mass of control characters makes it hard to read the text. This viewer seems to work round that by converting the document to plain text then viewing it (that's only my guess at how it works).
The viewing screen lets you have word wrap on or off, so you can get a quick view of the first part of each line or a more complete view with wide lines or paragraphs wrapped onto the next line. You can scroll through the text by moving the pointer onto the text and pressing space (or left mouse button) to scroll forward one line, or pressing enter (or right mouse button) to move forward by a screenful of text. There seems to be no way to scroll back through the text, although you can use the refresh symbol at the top to go back to the start of the text and work your way through again.
Actually, the term "wrap" is slightly misleading. I would have expected a word wrap to break lines at the spaces between words, although I think the author may just mean line wrap here, as all it does is to break the paragraph across two or more lines without trying to word wrap as such. And using the WRAP command doesn't actually change the view, you have to click on WRAP then on the REFRESH icon to update the display. You can compare the two different views in figure 2 and figure 3.



```
Copenhagen, Denmark, July 25, 1995

Dear QL user,

Thank you for getting a copy of PSION XCHANGE, the extended version of the

XCHANGE was programmed for the THOR based on the standard PSION QL software

Around 1989 the production of the THOR computers were stopped due to cash f

In early 1993 we contacted Hellmuth Stuven (former owner of Dansoft) to get

The THOR XCHANGE contained a minor degree of software protection against le

Improvements to Xchange

In addition to removing the copy protection to be able to release Xchange i
```

Figure 2

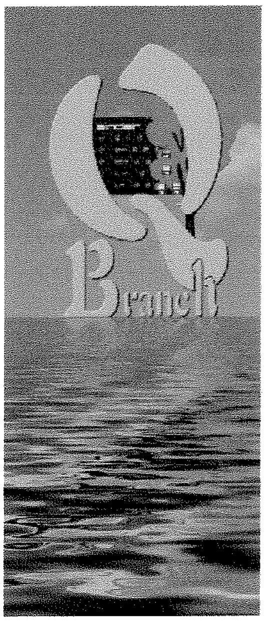
A small oddity is that I started off viewing a file called FLP1_README_DOC (which is actually from the Xchange program I'm using to write this). The filename is shown at the top of the screen - interestingly this is shown as RAM7_FLP1_README_DOC, so the program is obviously using ramdisk RAM7_ to store a plain text copy of the file it's showing on the screen (so the program needs a ramdisk, although most systems have one nowadays). The program does delete this temporary file after you've finished viewing.



```
Copenhagen, Denmark, July 25, 1995

Dear QL user,

Thank you for getting a copy of PSION XCHANGE, the extended version of the
original four classics.

XCHANGE was programmed for the THOR based on the standard PSION QL software
. Dansoft paid a considerable amount of money to have an improved version o
f the standard software but based around the concept of the IBM XCHANGE ver
sion. The software was then distributed on license from PSION as the standa
rd software running on the CST THOR PC computer. Three versions were made:
an english version, danish and a QUILL-only XCHANGE version in danish (sorr
y QL-users!).

Around 1989 the production of the THOR computers were stopped due to cash f
low problems. Dansoft and THOR International stopped the THOR business a ye
```

Figure 3

I have only three small gripes about this program. One small disappointment was that the program does not allow a filename to be passed to it in an EX command, so it cannot be used as a Fileinfo 2 doc file viewer, for example.
Another was that the program display cannot be resized for those with high resolution displays.
And finally, I wish it was possible to scroll backwards through the text without having to go back to the start of the file and work your way forward, which is a long winded way of doing it.
Apart from those small moans, this is an excellent little viewer which performs one well defined function extremely well. It's quite a small program, only 20 or 30KB long depending on which version you use. There's only a small text file supplied with the program, but it's so easy to use you don't really need much instructions.
This program is free and you can get it from most PD libraries and from websites such as those of Thierry Godefroy and Dilwyn Jones.
I got my copy from Thierry's site, the General Utilities page. From this site, it downloads as a file called docview100.lzh (the 100 indicating version 1.00), so you need a QL LHA archiver or Archivers Control Panel to decompress it if you get it from Thierry Godefroy's site. On Dilwyn Jones's website, it's on his Editors & Viewers page, as a zipped file which is slightly longer, so will take slightly longer to download.
Thierry Godefroy's QL software repository is at:
http://thgodef.nerim.net/smsq/
Dilwyn Jones's site is at:
http://www.dilwyn.uk6.net/index.html



Feeling out on a Limb?
Reach out for QBranch
Suppliers of Quality QDOS/SMSQ products
Hardware and Software

20 Locks Hill
Portslade
Sussex
BN41 2LB
UK
Tel: +44(0)1273 386030
Fax: +44(0)1273 430501
Mobile: +44(0)7836 745501
email : sales@qbranch.demon.co.uk
www.qbranch.demon.co.uk

We can accept payment by Visa, Mastercard, Maestro and Switch. We also accept sterling cheques drawn on a UK bank. or payment direct to our bank:
Natwest
Sort code 60-17-01
Acc. 84534826
BIC : NWBKGB 2L
IBAN :
GB34 NWBK 6017 0184 5348 26

QDT
The Ultimate QL Desktop
v1.02
£41.50 + p&p
£2.00 UK / 2.50 EU / £ 3.00 RoW
Needs SMSQ/E & extended colours to run

EasyPTR v4
Pts 1&2
Major Rewrite!
£42.00
Produce High Colour pointer-driven menus from SuperBASIC
Also on CD Rom

Text 87 £ 79.00
High Colour Patch (all systems) £9.00
ESC/P2 drivers £ 26.00 / Fountext £39.00
Typset94 / 2488 drivers £29.00 ea.

SMSQ/E and QPC 2

SMSQ/E for Gold Card / Atari / QXL / Q40
Aurora High Colour Driver included with Gold Card Version
£ 27.00

QPC 2 v 3.33
Now only £ 42.00

Upgrade Prices:
from v 3.xx Free
from v 2.xx £13.90
from v 1.xx £34.00

Delivered on CD ROM with Manual as Adobe Acrobat File

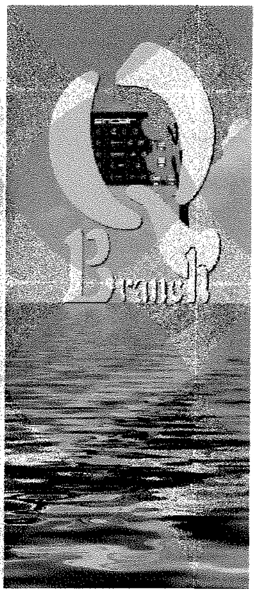
QPCPrint
Print to any Windows Installed Printer from Your QL Programs !
£27.00

Programs upgraded to high colour version
QMake / WinEd / QPAC 1 / QPAC 2 / QSUP / SuQcess / Disk Mate 5
Send old master disk and £1.60 per disk to cover costs
Free !

Paid Upgrades	
QD 98 to QD 2003	£10.50
QSpread 2001 to QSpread 2003	£10.50

Return Master Disk with order

There are a few items left from the Autumn clearout.
The following items are offered at sale prices:
GAMES : QShang/Firebirds/Pipes/Mine Field Arcanoid
(4/8 Colour only) £5.00 each + Postage
Black Knight Chess program / Super Games Pack £15.00 each!
All the keyboard membranes have now gone but, if you still need one. Contact Rich Mellor of RWAP.
It is a bit early I know but still -
QBRANCH Wishes you all a Merry Christmas and Happy New Year



Programs

Utilities

Fifi2	File Finder	£21.00
QSup	Utilies	£30.00
QSpread	Spreadsheet	£51.00
Cueshell 2	File Manager	£15.00
QPAC 2	File Manager & Utilities Package	£42.00
QPAC 1	Calendar, Clock, Calculator, Sysmon	£22.00
QLoad/Qref	Fast load for QDOS	£15.00
QTYP2	Spell Checker	£31.00
QLQ	Printer Utility	£30.00
SuQcess	Database	£28.00
Q-Route	Route Finder	£25.00
Knight Safe3	Backup Program	£35.00
Fractal Collection	Fractals	£35.00
QCount	Accounting	£25.00

Programming

QD 2003	Text Editor & More	£ 49.00
QBASIC	QLiberater to QD Link	£ 15.00
QLiberator	Basic Compiler	£ 50.00
QD + QBASIC		£ 63.00
QD + QBASIC + QLiberator		£104.00
QPTR	Pointer Toolkit	£ 32.00
MasterSpy	Fast Text Editor	£ 30.00
QMake	Assembler Tools	£ 18.00
QMon/Jmon	Monitor - Upgrade only	£ 22.00
BASIC Linker	Basic Library Linker	£ 22.00
Disa 3	Dissassembler	£ 34.00
QMenu	Menu Extensions & tutorial	£ 16.00
Easyptr v4	Toolkits & Programming Extns	£ 41.50
Easyptr v4	Part 3 C extensions	£ 14.00

HARDWARE

We have a rotating stock of both new and second user hardware. It is best to call or email us for details of what is available.

Recycled Items
(when available)

Super Gold Card	£110.00
Gold Card	£ 45.00
Aurora	£ 65.00
QXL	£ 35.00
superHermes	£ 65.00
DiRen Keyboard Interface	£ 15.00
Qplane	£ 5.00

New Items

Aurora	£70.00
Aurora w/8301 & Minerva	£80.00
8mb Rom Disq	£98.00
4mb Rom Disq	£65.00
2mb RomDisq	£39.00
mPlane	£34.00
MCplate	£ 6.50
Various braQuets	£ 8.00
Gold /Super Gold Card Batteries	£10,00

We also have a collection of standard QLs, QL Power supplies and some QL books.

Cables for the Aurora, Qubide and Super Gold Card ROMs and other QL accessories are also available from us.

Call for details

Month Number

by Dilwyn Jones

The QL clock is an excellent piece of programming - a simple PRINT DATE$ command will give you the time containing everything from the year number to the seconds in one short meaningful string, and if you use SMSQ/E it can even use any of the supported languages, e.g. English, French, German, Italian and any other language modules included.

```
2006 Sep 21 16:39:06
```

By copying DATE$ to a temporary string variable we can look at the information and extract the year, month, day, hour, minute and seconds information.

```
t$ = DATE$
year = t$(1 TO4)
month$ = t$(6 TO 8)
day_number = t$(10 TO 11)
hour = t$(13 TO 14)
minute = t$(16 TO 17)
seconds = t$(19 TO 20)
```

But the month details are returned as a name, not as a number, so it is harder to use in situations where you need to numerically manipulate time details. When I was writing a diary program recently, I needed the current month value as a number, not as a name, so I did some thinking and picked the brains of some of the people on the ql-users mailing list and after several suggestions I came up with enough routines and material to do the job, so I thought I'd write a short article about it. I realise that no programming is ever so good that it can't be improved, so I'm sure some of you will have even better ideas to contribute!
Although the intitial requirements are easy to grasp, the article goes into some depth about the programming involved. The routines can be used without having to understand them, although I'm sure the more experienced readers will appreciate the notes which accompany the listings. QL Today published a very useful set of articles some time ago dealing with date and time programming techniques ("Clocking In" Vol 6 Issue 5).
Assuming we are using an English QL system, the following routine is a simple starter, although unsuitable for SMSQ/E systems using other a language modules:

```
t$ = DATE$
month = t$(6 TO 8) INSTR
"JanFebMarAprMayJunJulAugSepOctNovDec"
month = (month DIV 3) + 1
```

This returns the month as a number from 1 (January) to 12 (December). This can be slightly improved to avoid the need for the "+1" part, as I saw when I read Rich Mellor's SuperBasic Reference Manual by adding a couple of dummy characters to the start of the month names list so that the found position DIV 3 returns a value from 1 to 12:

```
t$ = DATE$
month = t$(6 TO 8) INSTR
"..JanFebMarAprMayJunJulAugSepOctNovDec"
month = month DIV 3
```

This simple routine is perfectly adequate for simple programs which only need to use English month names, but for my program I wanted a routine which could return the details correctly on SMSQ/E systems which used another language module. So, after some correspondence on the ql-users mailing lists, a more useful routine was developed which allows us to extract the month names from the system for comparison.
This relies on the fact that the QL clock is based on a system which counts in seconds from midnight on 1st of January 1961. You can see this by using the DATE$ function with a seconds parameter of 0:

```
PRINT DATE$(0)
```

This displays:

```
1961 Jan 01 00:00:00
```

which is, conveniently, a sunday.

This leads us to a convenient way to extract the month names from the system by stepping along the first year in monthly blocks (anywhere in the month will do - multiples of 31 days will fall nicely somewhere in each month) and using DATE$ to extract the 3-character month names one at a time:

```
month_names$ = ""
seconds_per_day = 24*60*60
```

```
FOR a = 0 TO 11
  t$ = DATE$(a*31*seconds_per_day)
  month_names$ = month_names$ & t$(6 TO 8)
END FOR a
```

We can do a similar thing for day names if we wish to extract the language-dependent day names from the system, as I needed to do for the column headings in the calendar part of my diary program (with due credit to the QL Today Clocking In articles which enabled me to write the calendar generation routines in the first place):

```
day_names$ = ""
seconds_per_day = 24*60*60
FOR a = 0 TO 6
  t$ = DAY$(a*seconds_per_day)
  day_names$ = day_names$ & t$
END FOR a
```

Using the above information we can write a fairly short function to return a month number from 1 to 12 for the current month, allowing for the language dependent month names:

```
DEFine FuNction Month_Number
  LOCal
  month_names$ = ".."
  seconds_per_day = 24*60*60
  FOR a = 0 TO 11
    t$ = DATE$(a*31*seconds_per_day)
    month_names$ = month_names$ & t$(6 TO 8)
  END FOR a
  t$ = DATE$ : t$ = t$(6 TO 8)
  RETurn (t$ INSTR month_names$) DIV 3
END DEFine Month_Number
```

Therefore, month = Month_Number will result in a number from 1 to 12.

During the mailing list correspondence, Robert Newson kindly sent me a routine to find the current month number using a very different approach. He has kindly given me permission to reproduce the routine, so here it is. It's based on calculating the month as a number based on the number of seconds that the given time is since a known starting date point. It seems to be a great piece of programming, and I don't pretend to fully understand it, so I'll include Robert's notes to try to explain it rather better than I could.

```
DEF PROC my_date(secs, dy, mn, yr)
  dy = INT(secs / 86400) + 306
  yr = INT((dy + .8) / 365.25)
  dy = dy - INT(yr * 365.25) + 31
  mn = INT(dy / 30.6)
  dy = dy - INT((mn + 3) * 30.6) + 92
  mn = mn + 2
  IF mn > 12 : mn = mn - 12 : yr = yr + 1
  yr = yr + 1960
END DEF

DEF FN my_mth_no(secs)
  LOC dy, yr, mn
  dy = INT(secs / 86400) + 306
  yr = INT((dy + .8) / 365.25)
  mn = INT((dy - INT(yr * 365.25) + 31)
      / 30.6) + 2
  IF mn > 12 : mn = mn - 12
  RET mn
END DEF
```

I quote here from Robert's email:
"my_date has been made a PROCedure which modifies its calling parameters as more than 1 variable has to be set:

my_date secs, day, month, year

secs	=	*number of secs since 01.01.1961 00:00:00 of date to extract*
day	=	*stored with day of month (1-31)*
month	=	*stored with month number (1-12)*
year	=	*stored with year (1961-2099)[1]*

[1] Not checked beyond 20 Sep 2006, but it should be correct. 2100 would be wrong (after 28.02.2100) as the proc/fn doesn't realise it will not be a leap year; for the original application for which this was written, this wouldn't really pose a problem (as it wouldn't reasonably be expected to be running in 96 years' time), however, a minor modification could take into account those non-leap years.
I could have set a string and made it a function, but the whole point of the exercise was that you were trying to extract the day/month/year from the DATE$ (if I understood you properly) and it would have been a bit silly to package them up in a string to be unpackaged again afterwards. An alternative would be to add the three elements together, eg:

*packed = (year - 1961) * 372 + (month - 1) * 31 + day - 1*

Then:

day = packed MOD 31 + 1
month = INT(packed / 31) MOD 12 + 1
year = 1961 + INT(packed / 372)

The my_mth_no() function is extracted from the my_date procedure and takes one parameter:

seconds since midnight 1 January 1961 and returns the month number of that date:

month = my_mth_no(secs) "

Now, I still didn't fully understand this, so here's some additional notes that Robert sent me later in an attempt to clarify matters:

"The way it works is by using a calendar based on day 0 = 1 March 1960. The "year" in this calendar runs from 1 March one year to 28/29 February - the leap day is added at the end of the year, every 1460 days. (4 years)

Having the leap day at the end of the year means that we don't have to worry about it when working out the day offset for any date in the year 1 March or later.

*As we're passed seconds, we divide by 86400 = 24 * 60 * 60, the number of seconds in a day and ignore the fraction. The passed date needs to be adjusted for our shifted year and as day 306 is 1 January 1961, an offset of 306 is added.*

*The current year is calculated from the base by checking out every 365.25 days. We require the first year to be for days 0-364 (as our leap year happens to be in the fourth year), but dividing by 365.25 would put day 365 into the first year, similarly with day 370 for the second and day 1095 for the third year; we thus need to add a small correction to the number of days before division: 3 * 365.25 = 1095.75, so we have to be a little larger than 0.75 but less than a whole day, hence (dy + .8). Once the relative year to the base year is found, the number of days until the start of that year is removed leaving the day offset within the year.*

The number 30.6 is useful in that its multiples, when ignoring the fraction bit, provide an increase of 30, 31, 30, 31, 31 which then repeats. This matches the number of days in each of the months in the year that need to be added to get to the start of the current month: 31 (March), 30 (April), 31, 30, 31, 31, 30, 31, 30, 31, 31 (February has 28/29 which then starts the next year and so we don't need it).

Adding 31 to the current day number corrects for the sequence of the multiples of 30.6 so that it matches the month sequence. Dividing by 30.6 then gives the month number in the shifted year.

*We need to subtract the number of days in the month(s) preceeding this one from the day offset in the year. Again, we need to shift the 30.6 sequence and the (mn + 3) provides this. However, we also have to correct for INT(30.6 * 3) = 91 extra days which are subtracted, so we need to add them back with a +91. This then gives us the day offset in the month, so we need ot add an extra +1 to get the day number; this is included with the extra days added back by using +92.*

Finally the shifted month is corrected for the real year by adding 2. If the resultant month is › 12 (ie a month past December, ie January or February), subtract 12 from the month number and add one to the year.

The year is corrected to the real year by adding the base year of 1960."

Robert also suggested a small change to the my_mth_no function by removing the need for the mn variable as the value of the yr variable doesn't need to be corrected for January/February:

```
DEF FN my_mth_no(secs)
  LOC dy, yr
  dy = INT(secs / 86400) + 306
  yr = INT((dy + .8) / 365.25)
  RETurn (INT((dy - INT(yr * 365.25) +
61.6) / 30.6) MOD 12) + 1
END DEF
```

Robert writes:

"I tested this "improvement" (obfuscation I'd call it) and it seemed to work. I originally had:

*RET ((INT((dy - INT(yr * 365.25) + 31) / 30.6) + 1) MOD 12) + 1*

but to cut out one set of parenthesis and an operation, I included the +1 before the MOD as an extra 30.6 to be divided by replacing the +31 with +61.6. I was dubious about this as during testing the original procedure and function I came up with a "curious" feature of the floating point routines (in the JS ROMs):

In the line:

*dy = dy - INT((mn + 3) * 30.6) + 92*

I thought I could remove the need for one set of parenthesis by expanding the multiplication to:

*dy = dy - INT(mn * 30.6 + 91.8) + 92*

However, the value of dy became erratic and wrong for some months. Checking out the value of the INT() for one of these dates (mn = 7) exposed the problem:

```
mn = 7
PRINT (mn * 30.6 + 91.8)
306
PRINT INT(mn * 30.6 + 91.8)
305
PRINT ((mn + 3) * 30.6)
306
PRINT INT((mn + 3) * 30.6))
306
```

306, in this case is the expected, correct value. So why does the QL exhibit this feature?

I think it's simply because .2, .6 and .8 are infinite as binary decimals (should that be binaries?) - like 1/3, 1/7, etc are in decimal. Multiplying 30.6 by 7 introduces a small error in the .2 that added with the error in .8 gives ".9999998" and not "1". The PRINT procedure displaying this value doesn't display all the digits (by default) and so rounds it, but the INT function strips it off:

```
PRINT (mn * 30.6 + 91.8) - 305
.9999998
```

Extended DATE Function

A useful extension of the DATE function was introduced by the Minerva ROM, and is also present in SMSQ/E. Normally, DATE returns the current time as the number of seconds since the initial date (1/1/1961) mentioned before. This extended version takes six parameters to convert a set of 6 date values (year,month,day,hour,minute, seconds) to the equivalent in seconds. Since the DATE$ function can convert a time expressed in seconds to the normal date string, this is the natural opposite function:

```
LET year = 2006 : LET month = 9 : LET day = 10
LET hour = 12 : LET minute = 30 : LET seconds = 0
PRINT DATE(year,month,day,hour,minute,seconds)
```

prints the value:1.441888E9, which is that time expressed as seconds.

Please note that this version of DATE only works on Minerva ROMs and SMSQ/E, it is not implemented in Sinclair ROMs like version JM or JS. See Rich Mellor's SuperBASIC Reference Manual for further details.

Postscript

At about the time this article was written, Marcel Kilgus announced that he had written some date handling functions to perform a similar function in SMSQ/E versions from 3.13 onward. These are straightforward functions which return an integer number indicating year number, month number, day of the month and day of the week:

YEAR%	Returns year number
MONTH%	Returns month number 1 to 12, January to December
DAY%	Returns day of the month number, 1 to 31
WEEKDAY%	Returns day of the week, 0 to 6, Sunday to Saturday

Per Witte has also released some extensions to provide similar facilities for QDOS systems, which are available from his website at:
www.witteware.com/knoware

I would like to express my gratitude to Robert Newson for the time he has spent in correspondence regarding this article - his attention to detail was greatly appreciated. My thanks also to Marcel Kilgus for providing me with advance information about these extensions in advance of the release of SMSQ/E v3.13.

Start Here - Part 6

Essential Information For 21st Century QLers

by Roy Wood

Loading Resident Procedures

Having set up the screen, the colour depth and some other working variables it is time to start loading some code. The programs and extensions needed to get the computer to do something. The first things to get set up are the extensions and toolkits that make the programs you want to use work. Not all programs come with extensions and some just expect that you will have the extensions ready loaded anyway so, if you are graduating from the 'disk in, reset, boot, use program, reset (repeat)' school it is wise to consult the manuals and look carefully at the

BOOT programs on the master disks you received when you bought the programs.
Some things are nearly always needed. If you are running Pointer Environment programs you will need HOT_REXT, PTR_GEN and WMAN but these are part of SMSQ/E already so DO NOT reload them in the BOOT file unless you are not using SMSQ/E.
They will not generate an error but they can overwrite a more modern version of the extension. The same goes for Toolkit II and some of the other older extensions. Lightning and Speedscreen, for instance are now redundant for SMSQ/E users and QLOAD is built into SMSQ/E.

So What do we need?
Well a good start is the QLiberator extensions you can do this with:
```
300 LRESPR WIN1_EXT_QLIB_RUN
```

The next thing I load is probably one of the most controversial QL programs, QPAC 2.
```
310 LRESPR win1_progs_QPAC2
```

I won't go into this too deeply here but later in the article I will try to show those of you who may not believe in it just how helpful it can be.
Another useful extension provides the ability to use Environment Variables. This can be obtained from many places and can be found of the free Qascade distribution.
```
320 LRESPR WIN1_EXT_ENV_BIN
```

You can also add the Menu Extensions, The THING extensions
```
320 LRESPR WIN1_EXT_MENU_REXT
330 LRESPR WIN1_EXT_THING_REXT
```

You can probably see a theme creeping in here. I am storing all the extensions in a Sub-Directory called WIN1_EXT_ . This is so I know where they are and so I can be sure that, when a new version comes out I can update just one file, even if several programs use it, and know I am updating the version that is being loaded. You can repeat this with any other extensions that you may need. Suqcess, for instance needs DATA_BIN and DBAS_BIN in order to work and Disk Mate 5 needs a file called Extensions_cde.

Next we come to the grey area. These things live in the hybrid world where they are half-extension half-program. FileInfo2 is case in point. You will need to load:
```
340 LRESPR WIN1_FI2_FileInfo2_BIN
```

to get it to work. Don't forget that loading this program direct from the distribution will not work. It does need some careful configuration. It is a very powerful addition to the QL if you use any of the File Managers such as DM5, QPAC2, Cueshell and the new one being developed for QDT. To find out more look at Volume 4 Issue 5 where it is covered in depth.
QTYP is another 'grey area' program. The spell extensions need to be loaded to get the program to spell check in QD and other programs but they contain many elements that are normally considered to be the domain of programs.
```
350 LRESPR WIN1_QTYP_QTYP_SPELL
```

There are others like this but I think you get the drift from these examples.

Finally we have programs loaded as Resident Procedures such as QD, DataDesign, and FiFi. All of these can be LRESPR'd in the same way and will launch merely by typing a command like EXEP QD at the command line. The advantage here is that you only have one copy of the code running in the system no matter how many times you actually start the program.

One last area in which LRESPR comes in handy is in loading things like new system sprites. If you are using one of the new High Colour variants of SMSQ/E you can download some of the sprites Marcel Kilgus has put on his website and these can be added to your system using
```
360 LRESPR Win1_marcelth_syssprites
```

If you are using some versions of the ZIP and UNZIP utility you will probably also need to load the signal extensions. These are freely downloadable and can be found on the QTPI terminal distribution. They are used by the UNZIP program in particular to display the contents of the ZIP archive without actually decompressing it. The main program will work without it but it does throw up an error message if it is not there.
```
370 LRESPR win1_QTPI_sigext30_rext
```

ProWesS
The PROGS software team in Belgium released a series of programs and a whole new windowing system some time ago. Most of this is now free to use. This also needs certain files loaded in the BOOT file. These include their own system library, dynamic link libraries (dll) and a whole set of global variables.
```
430 IF PWSDIR$(LEN(PWSDIR$))<>"_" THEN
    PWSDIR$=PWSDIR$&'_'
440 LRESPR PWSDIR$&"ext_dll_rext"
```

```
450 LRESPR PWSDIR$&"ext_syslib_rext"
460 LRESPR PWSDIR$&"ext_global_rext"
470 :
480 REMark SBASIC extensions can't be
    loaded in startup files
490 REMark so they have to be loaded here
500 :
510 LRESPR win1_pws_ext_PWbasic_rext
520 dd_err=0 : REMark to allow DATAdesign
    error trapping
```

Note here that the PWBASIC extensions have been loaded after the others because they need some of the libraries loaded in the first part to be present before they can start.

At the end of the LRESPR section, if you have loaded the Menu Extensions, it is worth adding the Command:

```
590 OUTLN
```

This provides a general Outline to the screen which is useful if you write your own FE. programs in BASIC or if you use some of the free ones. It allows the windows to be resolved in the correct places on the screen.
One other thing to consider is the use of REMark lines. A year or so after you wrote that BOOT file you may look at it and think - 'why did I do that/ put that there/add that line?' A REMark line will probably give you a clue.

Environment Variables

If you are using programs which need their configuration to be done by the use of Environment Variables, setting these at this point is probably a good idea. Remember that you will have to do this after you have actually loaded the environment variable extension (ENV_BIN - see above).
Environment Variables are used by some programs instead of config blocks and they pass information to the program. They do not have to be set in the BOOT file but, if you are using the programs regularly and using the same setting it is a good idea to do so. The advantage of Environment Variables over config blocks is that you can change them at the command line and therefore change the behaviour of the programs they relate to. This will only work if the program has not been loaded as a resident procedure or is not already running when the variable are changed. You cannot rely on the program re-reading the variable when you come to use it if you have changed the variable after it has been started.
One good example of the use of these variables in with the MClock program that comes with Qascade and indeed with Qascade itself. Qascade for, instance, needs to find its '_inf' or '_rc' files. These are the text files that they read when they start up and which configure the programs.

```
600 SETENV 'QASCADE_RC=Win1_Qascade_
    qascade_rc'
```

Tells the program that the text file (qascade_rc) can be found in the Qascade subdirectory on WIN1_
MClock is a neat program which displays the clock in a button on the button frame. This, however, is not all it does you can set up two environment variable to tell it to launch programs.

```
610 SETENV "MCLICK_1=win1_progs_calendar"
```

will tell it to launch the QPAC2 calendar program if you click on the clock button with the left mouse key. Similarly,

```
620 SETENV "MCLICK_2=Win1_QDIARY_qdiary_obj"
```

will launch Dilwyn Jones' new diary program if clicked on with the right mouse button.

QDT will also write a series of environment variables to the BOOT file when it installs to help it find various files it needs when it starts up.
Environment variables are very useful tools for configuration and are, in some ways better than using config blocks. The one drawback with them is that they are completely user constructed and a simple typo can leave a program non functional.

EXECUTE ME!

Following on from these it is a good idea to set up a few of the programs that you are going to need in any session. I have four of these :

```
630 EX 'win1_progs_Qeyes_cf'
640 EX win1_cuedark_CUEDARK
650 EX WIN1_Qascade_qascade_exe
660 EX win1_progs_Mclock
```

The first excutes the QEyes program, the next the Cueshell screensaver, next Qascade itself and finally the MClock I have just mentioned.
I do load a few more programs a bit later on in the BOOT file and the reason for this is that some programs want to be started before others. This is very much a trial and error thing. Loading these four here will give me access to Qascade if the BOOT file aborts and they are happy being loaded at this point. At the end of the BOOT file I load QDT and Sysmon but we will come to that later.

HOTKEYS & THINGS

At this point we come to the commands that make QPAC 2 such a useful program to have loaded. As you saw at the start of this section I loaded it with one line - just a simple LRESPR command. By itself it does nothing - the really useful part is what you can do with it once it is loaded. As I have said before it is not one program but a collection of utilities and keywords, all of which can make your life much easier.

One of the things that QPAC2 will give you is a 'Button Frame'. This is an assigned area where you can place small buttons which can be configured to start programs or other tasks. Some people think this is unnecessary and, With QDT, Dilwyn's Launchpad and Qascade it is probably an old fashioned way of managing your programs but QPAC2 will give you a lot more than just this and the Button Frame has other uses than this.

One aspect of QPAC2 is the use of THINGS and 'HOT_' commands. Say, for instance, you have a full screen program like Text 87 running and you want to have access to a window buried under it. You could use CONTROL-C and page though to it or you could add a line to the BOOT file like this:

```
700 ERT HOT_THING(CHR$(232),'Button_Sleep')
```

This means that all you have to do is to hold down the ALT key and then press F1 and the program will be minimised onto a small button. To wake it again just click on the button. While it is minimised you have access to everything else on the screen.

Another aspect is the other way around. Again, say you are using, Text 87 and you want to call up you button frame and start something from it. This line:

```
710 ERT HOT_THING('.','Button_Pick')
```

will immediately make the button frame to the top job just by using ALT/'.'

```
720 ERT HOT_PICK ('b','')
```

Will do the same for the BASIC Windows.

Already we have three very useful lines and uses of QPAC2. I do not intend to go into a big discussion of QPAC2 - you can find that in another issue, but I will give you a few more useful lines of code to illustrate how you can use QPAC2 to your advantage if you are one of those people who bought it and never got past the tutorial on the disk. It does have a lot of uses so it it is worth investigating if you have not used it.

```
800 ERT HOT_THING ('1','Files';'\dwin1_')
```

for instance will give you a directory of WIN1_ just by using ALT/1 and

```
810 ERT HOT_THING ('2','Files';'\dwin2_')
820 ERT HOT_THING ('3','Files';'\dflp1_')
```

will do the same for WIN2_ and Flp1_ I think you can see how to do the same for any device attached to your particular QL.

QPAC 2 will also allow you to load a program when you need it with a line like:

```
880 ERT HOT_LOAD1 ('c','Win1_crib_spell
    crib_obj')
```

This means that pressing ALT/c will pop up the Spelling Crib program.

You can also 'wake' programs that have been LRESPR'd such as:

```
910 ERT HOT_WAKE ('q','QD')
```

This is very useful because you can call QD to the top if it is buried. I use this a lot with LINE-design when adding text from the Scrap thing.

If you want to use the QPAC2 Buttons these can be added with a line like:

```
920 BT_HOTKEY '1','WIN1_'
```

This will put a button the screen with the text 'WIN1_' in it. When clicked on it will execute the HOTKEY you have set up for ALT/1 (see line 800 above)

```
920 ERT HOT_CMD ('X','QPC_EXIT')
```

will quit QPC2 if used when the BASIC windows are open but HOT_CMD will work with any command that can be typed in the SuperBasic channel. This will only work if the BASIC Windows are active at the time and not buried or minimised.

```
950 ERT HOT_KEY ('Q',CHR$(240)&'Q'&CHR$
    (10))
```

This is another useful line that is the equivalent of pressing F3 - Q - ENTER which is the sequence many QL Pointer programs use to quit.

As we draw to the end of the BOOT file we will need to issue

```
960 HOT_GO
```

to initialise the whole HOTKEY system in QPAC2 and I add some further lines to my QPC2 BOOT file:

```
970 EW PWSDIR$&"prg_loader";PWSDIR$&
    "startup"
```

to start the ProWesS system

```
980 EX 'win1_progs_sysmon'
```
starts sysmon

```
990 EX win1_QDT_BIN_QDT_EXE
```
starts QDT

```
1000 QPC_SYNCSCRAP
```
synchronises the scrap function to the Windows clipboard - everything sent to scrap goes to the clipboard and vice versa

```
1005 HOT_DO 'b'
```
picks the BASIC windows to the top and
```
1010 HOT_DO CHR$(232)       :REMark put the
                             system to sleep
```
puts them to sleep on a button, leaving me with a clean screen.

And that is how I set out my BOOT. As you can tell from the gaps in the numbering there is a lot of other stuff I have left out but that is mostly repetition. There is more to QPAC2 as well and, if you want to get more from this program check vol 5 issue 6 where there is an in depth discussion of its properties.

I hope that this part of Start Here has led you to look at your BOOT files and I hope that some of you, who have maybe shied away from attempting to write your own BOOT files may now have been tempted to have a go. As I said before, unlike the PC system, you cannot do any lasting hard to your QL setup by writing a BOOT file provided you back up the old one and can put it back if you get the new one wrong. But then you all Back up your files don't you?

GPS and QPC – Part 1

by Hugh Rooms

Introduction

At the Hove meeting on May 28th 2006 I gave a talk about the U.S. Global Positioning System (GPS) and demonstrated the electronics and the SuperBASIC program I had written, under QPC, to display some of the data that can be received from it. I promised to write about it for both Quanta and QLtoday, and this is the result.
The QL interest is really just in the program but to use real time data from GPS some additional electronic hardware is necessary, so in this article I describe: how the GPS works; the hardware I use; and the program; with an account of the problems I encountered and the solutions that emerged.

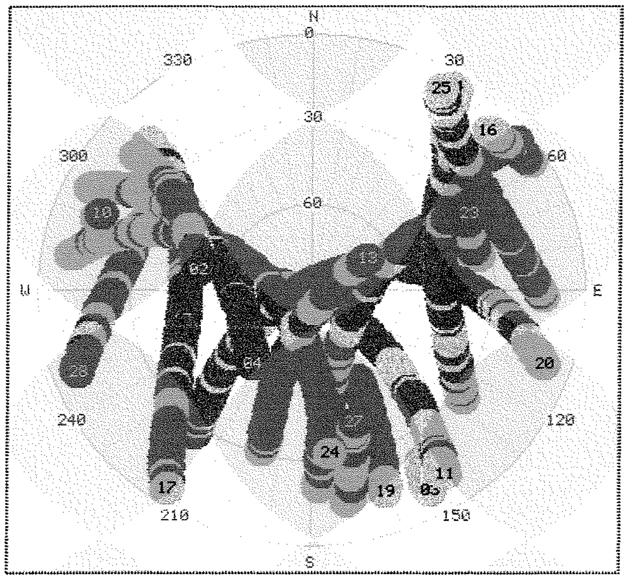


Fig. 5 - Orbits as 'sausages', over a five hour period

My program is run in one of several modes. If the hardware is available then it can extract data from the GPS receiver and display either the orbits of the satellites on a picture of the sky, figures 5, 6, 7 and 8, or, if it is loaded and run in a

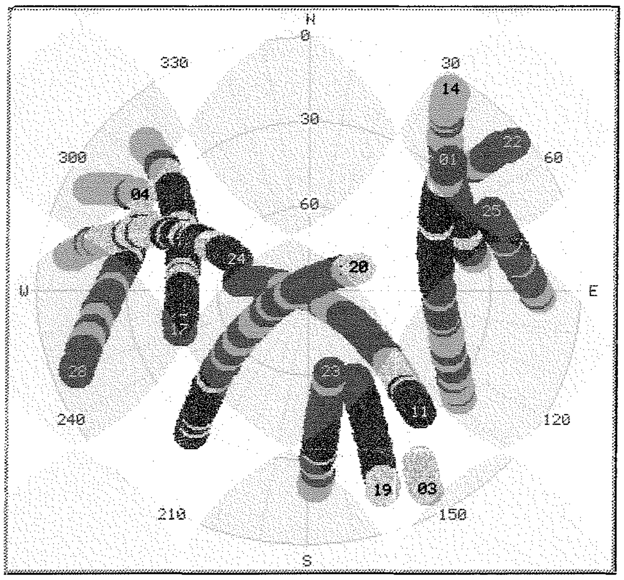
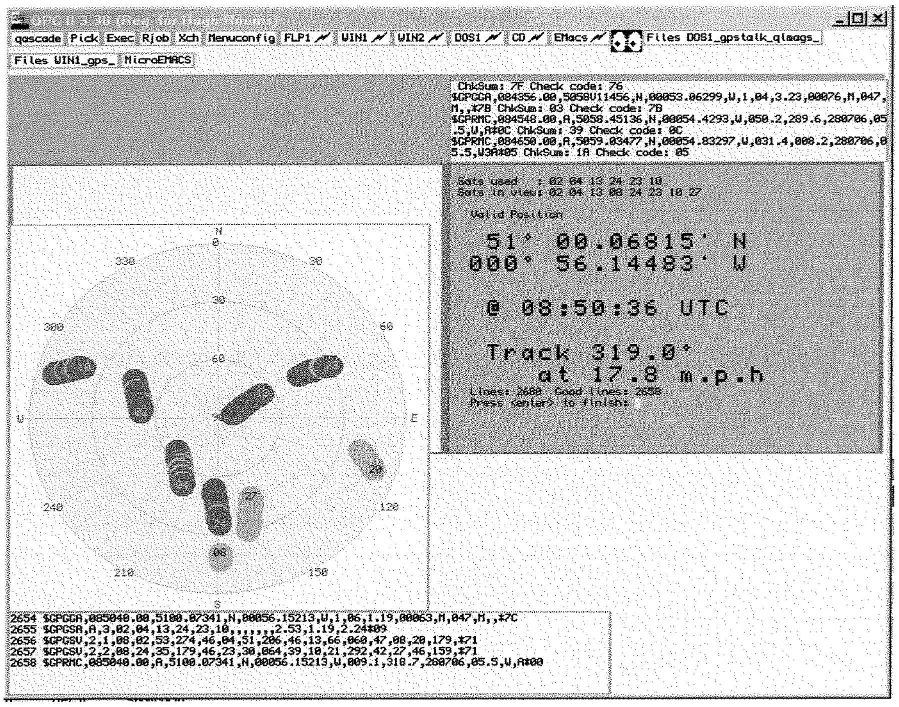


Fig. 6 - As figure 5 but interrupted after two hours or so to help make sense of it.

laptop and carried around, a plot on a chart of the track moved along as the journey progresses, figure 9. Don't get too excited -- the chart is a blank with only lines of longitude and latitude marked -- this is not a navigational system. Both sets of data, stationary or moving, can be saved, and viewed again as simulations. I hope to depo-



```
ChkSum: 7F Check code: 76
$GPGGA,084356.00,5058V11456,N,00053.06299,W,1,04,3.23,00076,M,047,
M,,*7B ChkSum: 03 Check code: 7B
$GPRMC,084548.00,A,5058.45136,N,00054.4293,W,050.2,289.6,280706,05
.5,W,A*0C ChkSum: 39 Check code: 0C
$GPRMC,084650.00,A,5059.03477,N,00054.83297,W,031.4,008.2,280706,0
5.5,W3A*05 ChkSum: 1A Check code: 05
```

```
Sats used    : 02 04 13 24 23 10
Sats in view: 02 04 13 08 24 23 10 27

 Valid Position

  51° 00.06815' N
 000° 56.14483' W

  @ 08:50:36 UTC

  Track 319.0°
      at 17.8 m.p.h
 Lines: 2680  Good lines: 2658
 Press <enter> to finish:
```

```
2654 $GPGGA,085040.00,5100.07341,N,00056.15213,W,1,06,1.19,00063,M,047,M,,*7C
2655 $GPGSA,A,3,02,04,13,24,23,10,,,,,,,2.53,1.19,2.24*09
2656 $GPGSV,2,1,08,02,53,274,46,04,51,206,46,13,66,060,47,08,20,179,*71
2657 $GPGSV,2,2,08,24,35,179,46,23,30,064,39,10,21,292,42,27,46,159,*71
2658 $GPRMC,085040.00,A,5100.07341,N,00056.15213,W,009.1,318.7,280706,05.5,W,A*00
```

Fig. 7 - Whole screen capture of an orbits display to show all the windows.

sit the program, with files of data, in the Quanta library, so you can experience its delights without having to buy a receiver and build the circuitry, or, indeed, without typing anything at all.

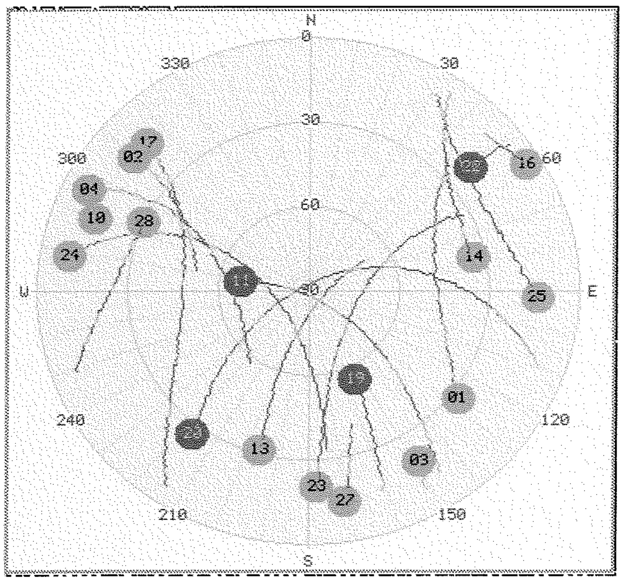


Fig. 8 - Orbits as 5 but with lines instead of sausages.

A caveat. Although I give this information in good faith, and as accurately as I can, it is a description of what I have done, not a recipe. If you want to try it out then you must make sure for yourself that what you do is correct. I have already blown up one receiver through carelessness: so be very careful.

Overview

GPS is a method of location using the calculated distances from a swarm of about two dozen artificial satellites, orbiting at around 12000 miles from the centre of the earth, in twelve hour orbits, and broadcasting data of time and position that allow a suitable receiver to calculate its own position in latitude and longitude, and its height above a datum. The orbits criss-cross in the sky so that any place on earth always has enough in view to get a good fix.

I've been fascinated by this ever since I first heard of GPS about ten years ago, but the receivers were too expensive at that time, and I am much more interested in how it all works than simply finding out where I am, so a complete but inaccessible unit wasn't really what I wanted. Now the prices have come down and you can buy hand held navigation equipment for under £200, or a bare receiver module for £32 (+VAT etc.) (1) (numbers in brackets refer to the Sources list at the end of this article). I bought a more expensive module (2) for about £60 (VAT etc made it £85) eighteen months ago and started experimenting with it. It is the second I have tried, I had blown up the first (3) through getting its connections wrong.

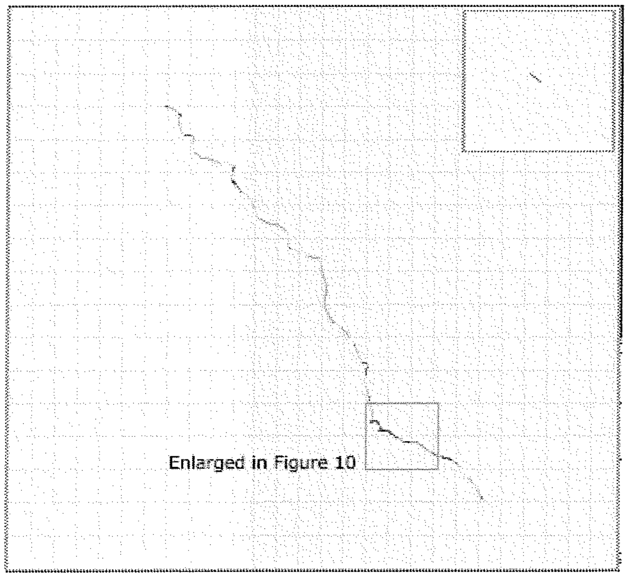


Fig. 9 - The track from Bognor Regis to Petersfield, this replaces the orbits window

RWAP Services NEWS!

We have moved back to Walsall for the time being - see our current address at the bottom of the page.

Better news is that we now have an extended range of second hand items for the Sinclair QL and ZX Spectrum, and have one remaining Epson 850 Inkjet printer available. Z88 stocks are however now getting quite low so if there is anything you need, please browse our website:

http://www.rwapsoftware.co.uk

We are also looking to produce some new hard disk interfaces for the ZX Spectrum and have a few little projects on the drawing board.

Our websites:
http://www.rwapservices.co.uk (General site)
http://www.rwapsoftware.co.uk (Sinclair computer second hand and new items)
http://www.rwapadventures.com (Adventure Programs)
http://www.internetbusinessangels.com (Guidance on setting up online businesses).

New Products!

QWORD 1.0 NOW WITH DIGITAL SOUND ON QPC2!

The wait is now over! Q-Word version 1 is finally available!

Platforms:
QPC/QXL, Q40/Q60, Aurora (with SGC)

Prices:

All versions without P-Word	£20.00
All versions with P-Word	£30.00

Notes:
Q-Word DOES NOT require SMSQ/E with GD2 support -OR- SMSQ/E at all on the Aurora or Qx0 machines. It works on the highest colour depth everywhere regardless of Operating System.
The Aurora version is available on either HD or ED disk. For the latter add £1.00 to the price. ED version is uncompressed and can be run directly from the floppy. All other Floppy versions are compressed. QPC/QXL version comes on CD. Non CD versions DO NOW support digital sound on QPC2

Quantum Leap ED **Drives**

The bad news is that our stock of ED Disk Drives has now been depleted and there is no sign of any more being available in the short term.

We do however have a range of brand new DD and ED 3.5" Diskettes for sale at low prices:

10 x 3.5" DSDD Disks £7.50
10 x 3.5" ED Diskettes £10
10 x 3" Disks £9

Prices do not include post and packing - please ask us for details.

We do have a range of second hand disk interfaces and drives (one or two ED Drives) for use with the Sinclair QL, so if you need anything, please let ask.

TALENT COMPUTER SYSTEMS for **Windows**

For QLers that run Windows or with incompatible hardware for Talent Games, we now have re-released these adventures so that they can run on your Windows-equipped PC. No Emulator, floppies, microdrive backups etc. required, just a one-click install! Of course the full QL line is still available! (See side column)

Talent Games for Windows ea. £ 10.00
(Each Game includes a runtime installation of QLAY-2 by Jimmy Montesinos)

Old Favourites!

Utilities

SBASIC / SuperBASIC Reference Manual on CD	£ 20.00
Sidewriter v1.08 *Landscape Printing (EPSON printers)*	£ 10.00
ImageD v1.03 *3D object generator*	£ 10.00
Q-Help v1.06 *Superbasic On-Screen help system*	£ 10.00
Q-Index v1.05 *Keyword-to-topic finder*	£ 5.00
ProForma ESC/P2 Drivers v1.04 for ProWeSs *Printer Driver*	£ 8.00

Applications

Flashback SE v2.03 (upgrade only) *Database*	£ 5.00
QL Cash Trader v3.7 *Accounting/Finance*	£ 5.00
QL Payroll v3.5 *Accounting/Finance*	£ 5.00
QL Genealogist v3.26 *Genealogy*	£ 20.00
Genealogy for Windows	£ 50.00
QL Genealogist to Windows version upgrade	£ 25.00
QL Cosmos v2.04 *Planetarium*	£ 5.00
Q-Route v2.00 *Route Finding*	£ 25.00
Upgrade from v1.xx	£ 5.00
Britain map v1.11	£ 2.00
BIG Britain map (needs 2Mb) v2.03	£ 5.00
Various Britain Area maps (ask for details)	ea. £ 2.00
Ireland map v1.00	£ 5.00
Belgium map v1.01	£ 2.00
Catalonia map v1.03	£ 2.00
P-Word UK English Dictionary (500.000 words!) *Dictionary*	£ 15.00

Leisure

Return to Eden v3.08 *Adventure*	£ 10.00
Nemesis MkII v2.03 *Adventure*	£ 8.00
The Prawn v2.01 *Adventure*	£ 8.00
Horrorday v3.1 *Adventure*	£ 8.00
West v2.00 *Adventure*	£ 5.00
The Lost Kingdom of Zkul v2.01 *Adventure*	£ 5.00
All 6 games above	£ 25.00
D-Day MkII v3.04 *Strategy/War Simulation*	£ 10.00
Grey Wolf v1.08 *Graphical Submarine Simulation*	£ 8.00
War in the East MkII v1.24 (upgrade only) *Strategy/War Simulation*	£ 5.00
Open Golf v5.20 *Sports Simulation*	£ 8.00
QuizMaster II v2.07 *Quiz*	£ 5.00
Stone Raider II v2.00 *Arcade Game*	£ 5.00
Hoverzone v1.2 *Arcade Game*	£ 5.00
Deathstrike v1.5 *Arcade Game*	£ 5.00
Flightdeck v1.0 *Flight Simulation*	£ 10.00
All 6 games above (Open Golf, QuizMaster II,Stone Raider II, Hoverzone, Deathstrike and Flightdeck)	£ 28.00

Notes on Software requirements
The following programs have a minimum SGC card requirement: P-Word, Qword, Big Britain MAP for Q-Route

RWAP Services
26 Oak Road, Shelfield, Walsall, West Midlands WS4 1RQ TEL: +44 1922 691607 From the UK Dial: 01922 691607
Website: http://www.rwapsoftware.co.uk
Email: sales@rwapsoftware.co.uk

We Accept Payment using:
PayPal VERIFIED NOCHEX (Cheques in £ sterling made payable to R. Mellor)

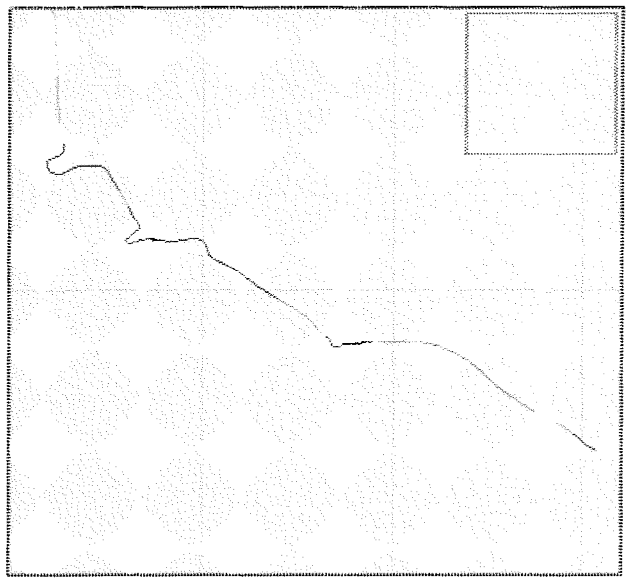


Fig. 10 - An enlargement of part of figure 9 to show the detail deep in the data.

My module, and as far as I can see most others, outputs its data in the form of ASCII text easily input to a computer for processing. As well as its current latitude and longitude – which is mainly interesting if you're on the move, once you've found out where you live – the module supplies the position (bearing and elevation) of each of the satellites in view, which of course changes with time, giving an opportunity for a changing display even for a stationary receiver. So my first objective was for a graphical picture of the sky and the satellites on it. This was the system I described at Hove. Since then I have gone on to use a laptop for a mobile system I can take in the car to display the speed, heading, and the track followed (also output from the module), with all the data stored in a file to view or use later, figure 9.
SuperBASIC is ideal for these applications, with its combination of hardware access, mathematics, and graphics, with a superb program editor (ED) giving quick and easy modification and testing.

GPS basics

Fairly basic equipment like mine gives an expected positional accuracy of ten to twenty metres. This in itself is truly remarkable considering that it represents the measurement of distances up to 20,000 kilometres to an accuracy of a few metres. With suitable, and much more expensive, equipment you can find your location (actually that of the antenna) to an accuracy of a centimetre or less -- even more remarkable. You now see surveyors on civil engineering and building sites using GPS instead of the traditional theodolite.

One thing I must get out of the way: if you search the Internet for GPS information you will see a lot of mentions of SA (Selective Access) in older documents. This was a security measure by the U.S. to prevent terrorists from using the service for really accurate fixes, called Precise Positioning System (PPS). Ordinary users were restricted to Standard Positioning System (SPS). SA was cancelled in 2000 and can now be ignored: anyone with the right equipment can get the maximum accuracy possible. Don't confuse this use of PPS, which is mentioned only here in this article, with Pulse Per Second which we will meet later on.
The satellites circulate along paths closely monitored and controlled by ground stations, and they have, of course, to be controlled to a greater accuracy than the position you are trying to calculate. Even so, at these accuracies, the gravitational pull of mountain ranges, and even weather systems, make the path more like that of the track of a bicycle about the general forward direction. So, to know where the satellites are, two sets of data are needed: firstly the basic orbital data, called 'almanac' in the jargon, which is stored in the receiver and is valid for several months at a time; secondly, detailed information about where the satellite is along this orbit, and any deviations that are going on, 'ephemeris', which changes frequently and is downloaded automatically from the satellites from time to time. I use the analogy of a train timetable: the printed version tells you what should happen – the almanac; but you also need detailed information, on the day, of cancellations, lateness, platform changes – the ephemeris.
The basic distance measurement is made by calculating the times the radio signals takes to travel from the satellites to the receiver, but it is difficult to make a very accurate clock in a reasonably cheap receiver so each satellite has a number of atomic clocks on board, again monitored and controlled from the ground and the receiver synchronises its own clock to GPS time. A pay-off from this is that you can get a very accurate time check – more than twice as accurate as the Rugby transmissions used in radio-controlled clocks.
Each satellite transmits its information over several minutes. At switch on, the receiver has no idea where it is, on the earth, above or below the surface, or way out in in space: it has to check that the almanac it has stored is current, or to read in a new one, and then get the ephemeris, a process that can take anything from a few seconds, if the 'off' time has not been too long, to an hour or so if it all needs to be read in.

To get a fix, four satellites are needed: the distance from the first defines a spherical 'surface' on which the receiver (strictly its antenna) must be, somewhere. The same applies to a second satellite, so the antenna is now known to be located on the circle where this 'sphere' cuts the first, since it must be on both. A third sphere cuts this circle in two places, and a fourth sphere passes through just one of these, giving a unique position relative to the satellites' frame of reference. Time data sent by the satellite are used at the receiver to make the calculations.
However these measurements and calculations are not precise, so the 'spheres' will not meet exactly at the same point. More satellites' distances are measured giving a spread of intersections of the spheres over a small space called a "resection". The size of the resection gives a measure of the additional inaccuracy called "Dilution Of Position" or DOP above that expected from the inherent tolerances of the system itself.

Where on Earth are we?

To know this we need to take the position obtained as described, relative to the satellites, and relate it to the Earth's surface, introducing a whole lot of new problems. Ideally we want our position in latitude and longitude, or some equivalent like an Ordnance Survey (OS) National Grid Reference.
Traditional marine navigation, giving a fix to a mile or so, can treat the Earth as a sphere, and use relatively simple trigonometry to calculate the position from a measurement of the elevations of the sun and/or stars at known times. The more precise GPS fix, and indeed mapmaking like the OS which aims at at least a two metre accuracy over the whole country, needs a model closer to the true shape of the Earth. However the Earth is rather bumpy, so it is 'Mean Sea Level' (MSL) that is taken as the 'surface' to work on: this real shape is called a "geoid", which, idealised to a lesser degree of accurate fit, becomes an 'ellipsoid' – a mathematical shape made by rotating an ellipse round its minor axis, in this case to make a 'bowl' shape (as in the game of bowls, i.e. not a 'rugby ball' shape). The great advantage of this model is that it is easily defined by just a few numbers, or parameters. The internationally agreed ellipsoid is WGS84 (6) and most GPS receivers give a position on this as the basic fix. As in all coordinate systems it needs an origin, which is basically the centre of mass of the Earth. The deviation of the ellipsoid from a sphere, the 'eccentricity', is only about 40 kilometres, or 0.3% of the Earth's average diameter of 12800ish km. It is all a deep and complex subject, well explained in (4).
There are a lot of 'howevers' in all this, and the next is that, although WGS84 or other global ellipsoids are relatively easy to calculate, none is close enough to the measured MSL everywhere (anywhere?). This was realised early in the nineteenth century when the OS first surveyed the country. The then Astronomer Royal, Sir George Airy, calculated a better local ellipsoid reference for the UK, now called 'Airy 1830', which was used for the OS. In modern times, Airy 1830 has slightly different radii, origin, and eccentricity from WGS84, but again the parameters are few and easily converted (in all this 'easy' means compared with the enormous mass of data and calculation necessary if a more detailed model was used; the maths looks pretty horrendous in any case.)
There's a good story here (8). One axis for Airy 1830 uses, of course, the Greenwich Meridian. Originally defined by the position of the transit instrument built by the first Astronomer Royal, Flamsteed, in 1685 when the Greenwich Observatory was established to gather astronomical data for marine navigation, in 1725 the meridian moved 73 inches East to where the next Astronomer Royal, Halley, built his transit when he discovered that Flamsteed's instrument was subsiding out of alignment. With the accuracies expected at the time the small change in position did not matter. In 1750 Bradley built a more modern transit a further 436 inches East and the OS started with this as its datum; this instrument still exists at Greenwich in its original observing room. In 1850 Airy was after even greater accuracy and built the present transit at 19 feet East of Bradley's, sending a memo to OS to inform them of the change in the Meridian. The brass strip showing this meridian for tourists was added in the 1970s.
Unique among major countries, the UK was resurveyed in the twentieth century, from about 1938 to 1979, to correct anomalies that more accurate surveying techniques had revealed. The new survey followed the earlier one, from a base line across Southern England, with all other points fixed by measuring angles. As a check a second 'base line' across Scotland was measured, and came out well within the accuracy expected. But the new survey placed the 'brass strip' Greenwich Meridian at 00 degrees 00' 00.418" E longitude – an 'impossible' result, 26.4 feet out. What had happened was that the first

Figure 4 - Ten lines of typical data

```
$GPGGA,111656.00,5047.66176,N,00040.25166,W,1,05,1.73,00015,M,047,M,,*79
$GPGSA,A,3,01,23,11,20,28,,,,,,,,2.65,1.73,2.00*0C
$GPGSV,2,1,08,01,46,072,43,17,42,297,29,11,66,130,35,20,74,241,35*70
$GPGSV,2,2,08,14,16,037,32,23,26,177,33,24,33,291,26,28,10,251,32*76
$GPRMC,111756.00,A,5047.66170,N,00040.25170,W,000.0,134.1,240406,05.4,W*07

$GPGGA,111758.00,5047.66170,N,00040.25171,W,1,06,1.35,00016,M,047,M,,*74
$GPGSA,A,3,01,11,20,23,14,28,,,,,,,2.00,1.35,1.47*08
$GPGSV,2,1,08,01,46,071,43,17,43,297,,11,65,130,34,20,74,242,35*78
$GPGSV,2,2,08,14,15,037,32,23,26,177,32,24,34,291,,28,10,251,33*76
$GPRMC,111852.00,A,5047.66167,N,00040.25172,W,000.0,344.3,240406,05.4,W*0F
```

Note: the empty line is not sent by the GPS module, it is inserted to make it easier to see what is going on.

OS had 'forgotten' about Airy's memo and continued with Bradley's meridian. When this was taken into account the discrepancy was well within the allowed tolerance. However all OS maps continue to be based on Bradley's Meridian, over eight metres East of what you see as 'The' Greenwich Meridian if you go to the Observatory. At the accuracies of GPS, tectonic drift of the continents is apparent, and they all move in different directions (11). Based on an average over the world, Europe including UK is moving NE at about an inch a year, almost a metre since WGS84 was established. To avoid the continual updating of map data, a reference called European Terrestrial Reference System (ETRS89) is used that is fixed to, and moves with, Europe, with transformation data to and from WGS84 provided by national organisations.
The WGS84 meridian and origin are based on an average of the world, made so that the transformations from local datums is on average as small as possible. On this system the Prime Meridian, on which all GPS is based, is a further 334 ft East of Airy's, and not marked with a brass strip as far as I know.
Further complications arise because latitude and longitude, on the near spherical Earth, are curves, whereas National Grid lines are straight lines on a flat map. (6 page 35) Yes, I know it's really a cylindrical map, but that is another story.
The net result is that a GPS WGS84 Latitude and Longitude plotted on an OS map places you at roughly 120 metres NWxW of your true position – I've seen various estimates, but this looks the most likely, I took it from the plan of Southampton University on page 4 of (6), and it fits quite well with the figures I've given for the Meridian. Your GPS device may do the translation for you, but it is something to check.

GPS data

From the data sent from my receiver to QPC, I can display, as alternatives, the track across ground, figure 9, which I use when carrying the equipment in the car; or the orbits extending across the sky, figures 5, or 8; a nice picture to watch at when at home with the receiver stationary for a long period.
The receivers available, and I have studied only the two I've owned, do a tremendous amount of processing on the data received, and output the results as a stream of ASCII codes. The data format is specified by the USA National Marine Electronics Association (NMEA)(9), in the form of 'sentences'. Each starts with a '$' sign and an identifying string, and terminates with a check sum and ‹CR›‹LF› codes. My present module outputs a default set of five sentences every second, seen in figure 4 and at the bottom left of figure 7. The data sheet for the device says that it can be programmed in a number of ways to change its behaviour, but I have not investigated that, and won't mention it again.
My hardware converts and sends this data as RS232 signals to the PC.
A typical sentence is the first one sent each second:

```
$GPGGA,123519,4807.038,N,01131.000,E,1,
08,0.9,545.4,M,46.9,M,,*47
```

Which, noting that data fields are delimited by commas, translates as:

`$GP`	Header code for all sentences
`GGA`	Global Positioning System Fix Data: Name and type of data
`123519`	Fix taken at 12:35:19 UTC
`4807.038,N`	Latitude 48 deg 07.038' N
`01131.000,E`	Longitude 11 deg 31.000' E

Note: degrees: minutes and decimals of minutes are run together!

1	Fix quality: 0 = invalid 1 = GPS fix (SPS) 2 = DGPS fix 3 = PPS fix 4 = Real Time Kinematic 5 = Float RTK 6 = estimated (dead reckoning) (2.3 feature) 7 = Manual input mode 8 = Simulation mode
08	Number of satellites being tracked
0.9	Horizontal dilution of position (I will explain this later)
545.4,M	Altitude, Metres, above mean sea level
46.9,M	Height of geoid (mean sea level) above WGS84 ellipsoid
(empty field)	time in seconds since last DGPS update
(empty field)	DGPS station ID number
*47	the checksum data, always begins with *

I must admit that don't know what a lot of this means, and at present I don't use this data at all except to recognise the start of each second's transmission.
To get it out of the way: the checksum, present in every type of sentence, is the 'XOR' of the binary of all code between '$' and '*', expessed as two hex characters. My function 'CheckSum' probably gives a better explanation than words would.
I found that data is often corrupted, and doesn't conform with the standard. It is also apparent that different module manufacturers interpret the NMEA standard in different ways (surprise, surprise), particularly in the number of characters in each field. So a considerable amount of checking is needed to eliminate potentially misleading data. My program doesn't always do the checks – a combination of "couldn't be bothered (CBB)" and "It Works So Why Fix It? (IWSWFI)". That is: I often do rely on the data being in the format expected. Where this does result in corrupted data further checks usually cut it out. The last real field often ends at the '*' that denotes the checksum, instead of a comma; until I realised this and allowed for it (in the 'field$' function) it caused a lot of unexplained crashes.
There are many more of these sentence forms defined, but just four types, including $GPGGA, are sent each second by the device I use, (one type, $GPGSV, is sent twice to make the five mentioned, as I will explain). The other types, in the sequence they are sent, are:

```
$GPGSA,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39
```

Where:

GSA	Satellite status
A	Auto selection of 2D or 3D fix (M = manual)
3	3D fix - values include: 1 = no fix 2 = 2D fix often happens if trees etc spoil the signal 3 = 3D fix only this really is a 'good' fix
04,05...	PRNs (identification numbers) of satellites used for fix (space for 12)
2.5	PDOP (dilution of precision): this is horizontal and vertical combined
1.3	Horizontal dilution of precision (HDOP)
2.1	Vertical dilution of precision (VDOP)
*39	the checksum data, always begins with *

A 3D fix is the best, with good visibilty to at least four satellites. If only three are available then it is still possible to get a fix using the line to the origin of co-ordinates by assuming an elevation above where it cuts the ellipsoid. If the ellipsoid was a true sphere, then the height would not matter, but, with eccentricity, the height assumed must be close to the true height for a '2' fix to be accurate, so really they are only any use near sea level. The question of what "height" is, is gone into at great length in (6).
PRNs – "Pseudo Random Noise", and no, I don't understand this; for more information see (10).
The DOPs I have mentioned earlier, and this sentence quantifies them. The range is 1 (perfect position) to 50 (completely lost). Anything under 3 is pretty good. Vertical (VDOP) is always the worst, as GPS is intended primarily to give a horizontal position.
From the $GPSGSA sentence I extract the identifiers of the satellites actually used to derive the position at this instant.

```
$GPGSV,2,1,08,01,40,083,46,02,17,308,41,
12,07,344,39,14,22,228,45*75
```

Where:

GSV	Satellites in view
2	Number of sentences for full data
1	sentence 1 of 2
08	Number of satellites in view
01	Satellite PRN number
40	Elevation, degrees
083	Azimuth, degrees, clockwise from North

46	SNR - (Signal to Noise Ratio?) higher is better, a measure of the quality of the signal.

The above four fields repeated for up to 4 satellites per sentence

*75	the checksum data, always begins with *

There are usually two $GPGSVs sent each second, listing all the satellites in view with their position in the sky, the basis of my sky display. From it I also get the "Satellites In View" list which includes those not actually used in the present fix. Curiously, the module I have misses out the SNR for the last satellite on each line.

Finally:

```
$GPRMC,123519,A,4807.038,N,01131.000,E,
022.4,184.4,230394,003.1,W*6A
```

Which translates as:

RMC	Recommended Minimum Content
123519	Fix taken at 12:35:19 UTC
A	Status A= Active or V= Void. A void line is not a reliable fix.
4807.038,N	Latitude 48 deg 07.038' N
01131.000,E	Longitude 11 deg 31.000' E

Note: degrees: minutes and decimals of minutes are run together!

022.4	Speed over the ground in knots
184.4	Track angle in degrees True, measured clockwise from North
230394	Date - 23rd of March 1994
003.1,W	Magnetic Variation, not always given: an empty field ",," is shown instead
*6A	The checksum data, always begins with *

I extract the current latitude and longitude, date and time, and the track and speed for the displays. Note that the GPS receiver can have no idea which direction it is facing, you can't use it as a compass: it can only take successive positions at known times and from them calculate track and speed.

The display in my set up is a second or two behind the real time of the data because of the processing time in the hardware and in QPC, and the track display is updated only after the program's 'Delay' time, so it tends to smooth out sharp bends in the track.

Greater accuracy

Looking at a latitude given in the $GPRMC sentence as, say, 51 degrees 43.26815'N, and remembering that one nautical mile is one second of arc on a great circle such as a meridian, we see that the 43 represents nautical miles in 1852 metre units; so the 2 is in 185 metre units, the 6 is in 18.5 metres, the 8 a bit under 2 metre units, the 1 is a unit of 18.5 centimetres, and the 5 in 18.5 mm units. We should expect those last few digits to be unreliable, and the best latitude for this reading from one receiver is 51 degrees 43.268'N, with the last digit a bit doubtful. For longitude, each minute is one nautical mile multiplied by the cosine of the latitude, about 0.63 for Southern England, or around 1100 metres per minute of arc, so the later digits of a longitude are even more unreliable.

The rest of this is what I understand from what I have read on the Internet, I've no direct experience of it to modify my impressions. A good read is (5).

I mentioned that centimetre accuracy can be obtained from GPS. As it is, in my set up, the accuracy is enough to just differentiate the two sides of a single carriageway main road. Bad data, e.g. from being under trees, misses some points out so it may not give a really true picture of a bendy road.

A better fix can be obtained, if you are stationary, by taking an average over a long period, and picking out the readings with a low DOP. I may revisit this to see what happens, but there's not much further I can go with the equipment I have.

To get a much higher accuracy a procedure called 'Differential GPS' is used. When observing the position reported by a stationary receiver, the decimal points of the seconds of arc 'dither', as seen on the display of part of a data file, figure 4. Successive calculations give slightly different results, and the device itself cannot know whether this is really due to the antenna making small movements, or to errors inherent in the system, at a stationary receiver, from small wobbles in the orbits and propagation conditions for the signal as it travels through the ionised layers of the atmosphere.

Now we can assume that two GPS receivers, not too far apart, experience exactly the same errors, with the true difference between their positions at the same instant being given by the difference between the calculated positions, with the errors cancelling out. 'Not too far apart' can, apparently, be 30 to 100km.

If one, reference, receiver is placed at an accurately known location as a control station, the 'wobbles' in apparent position there become known errors, so the other receiver, usually roving around an area being surveyed, can

TF Services

Compswitch

A UK 4-way trailing socket designed to switch off computer peripherals automatically when the computer is switched off, or (in the case of an ATX computer) when it auto-powers down. *Compswitch* has one control socket, and three switched sockets.
Can be used with lights/hifi/monitors—ie a QL monitor can be used as a switch control.

Cost £24

superHermes

A major hardware upgrade for the QL
All Hermes features (working ser1/2 at 19200, independent baud rates/de-bounced keyboard/ keyclick) IBM AT kbd I/F // HIGH SPEED RS232 at 57600// serial mouse port and 2 other RS232 inputs// 3 I/O lines // EEPROM

Cost (including manual/software) ....**£90** (£92/£93)
IBM AT UK layout Keyboard...............**£11** (£13/£15)
Serial mouse...**£8** (£8.50/£9)
Capslock/scrollock LED ....**£1** (£1.50/£1.50)
Keyboard or mouse lead ....**£3** (£3.50/£3.50)
High speed serial (ser3) lead.............**£4** (£4.50/£4.50)

Hermes available for £25 (£26/£27) Working ser1/2 and independent input, debounced keyboard.

SuperHermes LITE: All Hermes features (see above) + an IBM AT keyboard interface only.
Cost (incl keyboard lead) ....**£53** (£54/£55)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 incl 6 month guarantee

Minerva

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...**£40** (£41/£43) MKII...**£65** (£66/£67)

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I^{2}C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

QL RomDisq

Up to 8 mbyte of flash memory for the QL
A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second.
Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq............**£39** (£40/£41)
4mbytes RomDisq..............**£65**(£66/£67)
8 mbytes RomDisq..........**£98** (£99/£100)
Aurora adaptor.....................**£3** (£3.50/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost ....**£34** (£35/£36)

I2C INTERFACES

Connects to Minerva MKII and any Philips I^{2}C bus

Power Driver Interface 16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)
2 amp (for 8 relays, small motors)....................**£40** (£43/£44)
4 amp total (for motors etc).......................**£45** (£48/£50)
Relays (8 3a 12v 2-way mains relays (needs 2a power driver)...**£25** (£28/£29)
Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required..........**£25** (£27/£28)
Analogue Interface Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temp measurements, sound sampling (to 5 KHz), x/y plotting.....................................**£30** (£31/£32)
Temp probe (-40°C to +125°C)..............**£10** (£10.50/£11)
Connector for four temp probes..............**£10** (£10.50/£11)
Data sheets...**£2** (£2.50/£3)
Control software & manual (for all I/F)........**£2** (£2.50/£3)

QL SPARES

Keyboard membrane ....**no longer on sale**
1377 PAL..**£3** (£3.50/£4)
Circuit diagrams..**£3** (£3.50/£4)
68008 cpu or 8049 IPC......................................**£8** (£8.50/£9)
8301/8302 or JM ROM or serial lead...........**£10** (£10.50/£11)
Power supply (sea mail overseas)....................**£12** (£19/£23)
Other components (sockets etc) also available

Prices include postage and packing (Airmail where applicable) Prices are: UK (Europe /Rest of world). Payment by cheque drawn on bank with UK address,/postal order or CASH! I can no longer accept card payments as UK only does PDQ transaction. SAE or IRC for full list and details 22 Feb 03

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: +44 (0) 1442-828254 Fax/BBS: +44 (0) 1442-828255
tony@firshman.co.uk http://www.firshman.co.uk

derive its own position very accurately by applying the 'wobbles' in reverse to its own readings. In practice this can be exploited in several ways: a succession of readings are taken at the reference point over a session, and the corrections collected during the day are correlated at the office in the evening; alternatively it can be done 'on the fly' using a radio link to send the corrections to the roving apparatus, the latter option being more expensive of course, but it does allow for continuous checking and immediate dealing with dodgy fixes without having to revisit the site. Either method brings the accuracy down to the metre region, and there are further refinements of technique if accuracies down to the centimetre level are required.
The OS now uses this method, and has surveyed a number of the old triangulation pillars to a very high accuracy, and additional control stations have been built, with the data available free on the Internet. These are 'Passive' stations -- you have to provide your own equipment. A numer of 'Active ' stations also exist, which broadcast the correction data to be picked up at a survey site in the appropriate area. Links to lists and details are in (4).

Letter-Box

Steve Poole writes:
In my letter on page 38 of QL TODAY, volume 10, issue 5, I complained of main-frame mentality reducing people to numbers. It was also suggested that readers could donate their wait-states to Science, for the good of the common cause.
Citizen Science began in 1998, when the SetiAtHome project was launched. SETI is the abbreviation for 'Search (for) Extraterrestrial Intelligence'. It is a University non-governmental research project, that uses the largest radio-telescope in the world, the 300-metre diameter Aricebo-crater dish in Costa Rica. This is a fixed dish, aimed at a small area of the sky, that searches all radio frequencies to try to detect any intelligent radio signals, amid the hiss of white background 'noise' emitted by distant pulsating stars.
SetiAtHome hopes to find some sign of intelligent life in Space, (as some philosophers would question the existence of any real intelligence on earth!). This project involves listening to thousands of wavelengths simultaneously, which demands enormous computing power. So the project organisers hit upon the original idea of popular 'distributed computing': Any member of the public could donate the idle-time of his computer to Science: First he must contact the website and register. Then the site will download a search-subroutine onto his computer, which would operate whenever the computer was just ticking over. Thereby, SETI has between 200,000 and 1,000,000 PC's running its program at any one moment. This amount of computing power is more than available on any super-computer. And the wavelength-search, which would have taken centuries on one research machine will be complete in just a few years. While your computer is thus occupied, the SETI routine displays its own screen-saver.
There are many PC owners who are extremely enthusiastic, and supply extra computers just to participate to a maximum. These are nick-named 'Crunchers', and some run simultaneously up to 15 or more different projects: These range from research on 'Aids' to 'Prime Numbers,' and new ones appear every month. Here is a selection:
ClimatePrediction.net ... Give a hand to the world's biggest computers to predict disasters.
EinsteinAtHome ... See if you spot a 'Gravity Wave' to prove the General Relativity Theory.
PrimeNumbersAtHome ... Get more than any other group to decode any hostile messages.
RosettaAtHome ... Make 3D stereo models of proteins for medical research.
StardustAtHome ... Scan photos of satellite gels to find the first Stellar-dust particle.
As you see, there are many ways to humanely reduce the whole of creation to numbers, just use your search-engine to uncover sites that interest you. Distributed computing is tipped to represent progress in the future. Already PC's have dual-processors, and progress will more likely be in the development of parallel programming rather than in miniaturising circuits, which may now have approached all reasonable limits. Indeed Giant Computers are mainly arrays of thousands of ordinary microprocessors, rather than giant processor-chips. Perhaps somebody in the QL community could provide software to link QPC's or even better QL's to compute in Parallel. A first start would be for someone to write an article on parallel processing on QL's! Any Offers?

More programs from Steve in the next issue.

George Gwilt writes:

Comments on Programming in Assembler - Part 15

by Norman Dunbar (QL Today Vol 11 Issue 1)

I am glad that Norman Dunbar has restarted his series on assembler programming.

In Part 15 he has code which converts a long word from binary to ASCII. Oddly enough, although he says that "reusing existing and working code is always a good idea" the conversion code is not written as a subroutine. Does this mean that he has not used this routine elewhere?

As it happens I found that I had to do this sort of conversion so many times that I put my code in a library and used it again and again. Since it is totally different from Norman's code I thought it might be of interest to show it.

My method is to successively divide the number by 10 until it becomes zero. Each remainder is the next digit, starting from the least significant. If n is the number, iterating

```
d = n MOD 10
n = n DIV 10
```

gives successive values of the digits d.

A complication arises if the number n is bigger than 10*2^16 because division by 10 will set overflow. We thus resort to multiple length division.

Suppose that the successive words of the number are p1, p2, p3 etc each being less significant than the last. Then we start with the pair 0 I p1. After division we have r1 I q1, where the quotient is q1 and the remainder r1.

We now use r1 as the the more significant word of the pair r1 I p2 and do a further division. We thus have:

```
0  | p1 -> r1 | q1
r1 | p2 -> r2 | q2
```
etc

At the end we have the last remainder as the digit we are looking for and we replace p1p2p3... by q1q2q3... and we are ready for the next digit.

In the present case, of course, we only have p1 and p2. The division of 0 I p1 takes place in D0 and that of r1 I p2 in D1

```
; LTOD sets ASCII decimal for D0.L in
; a buffer indicated by A0 (and uses D1).
;
; At entry:
;   D0.L contains the (positive) number (a | b)
;   where  a | b stands for the high and low words, so D0.L = a*2^16 + b
;   A0 points to the end of the area to contain the ASCII decimal.
;
; At exit:
;   D0 = 0
;   A0 points to the start of the number.
;
LTOD     SWAP     D0        ; b | a
         BEQ.S    LTODW2    ; If zero just set "0"
         MOVE.W   D0,D1     ; a to D1.W
         BEQ.S    LTODW     ; If zero only one division is needed per digit
         EXT.L    D1        ; a to D1.L (0 | a)
         DIVU.W   #10,D1    ; Divide by 10 ( r0 | q0 )
         SWAP     D1        ; q0 | r0
         MOVE.W   D1,D0     ; r0 to D0.W  (b | r0)
         SWAP     D0        ; r0 | a
         DIVU.W   #10,D0    ; r1 | q1 - no overflow since r0 < 10

; Now (q0 | q1) = (a | b) DIV 10 and r1 = (a | b) MOD 10

         SWAP     D0        ; q1 | r1
         ADDI.W   #48,D0    ; Set r1 to ASCII . .
         MOVE.B   D0,-(A0)  ; . . and store it
         SWAP     D0        ; Set q1 to D0.W
         MOVE.W   D0,D1     ; Now D1 = q0 | q1 ie the new (a | b)
         MOVE.L   D1,D0     ; Set to D0.L for next iteration
         BRA.S    LTOD      ; Loop
```

```
;
; At this point the number is < 2^16
;
LTODW1    DIVU.W   #10,D0    ; r | q
          SWAP     D0        ; q | r
LTODW2    ADDI.W   #48,D0    ; Set r to ASCII . .
          MOVE.B   D0,-(A0)  ; . . and store it
          CLR.W    D0        ; b | 0
LTODW     SWAP     D0        ; 0 | b
          BNE.S    LTODW1    ; If b <> 0 there is more to do
          RTS
```

Norman works forwards; I work backwards. Norman uses a table of powers of 10; I use division. The total number of instructions is similar if you count the Table as instructions. The timing will vary from number to number. My division will always take overall about 4 to 5 times as long as Norman's "digit_loop", but Norman's loop must be iterated from one to ten times per digit.

I cannot resist making five more comments on the program.
The first comment is on the code at "convert" on page 25. This is where the new dataspace value is entered by the user. The code will happily continue to take in numbers until "k" or "K" appears. There are two awkward cases. The first is where the user has entered more than 10 digits before the k. In this case the number in D5 calculated by the code at "mul_10" will be wrong because of overflow. The second case is where no k is entered. This is more interesting because after "try_next" the code will drop through to "make_even" which will assume that the amount has been multiplied already by 1024. However, the next piece of code will ensure that at least the minimum space will be set. It might be kinder to the user to check that D5 does not overflow and that you do not drop through "try_next".

These errors could be trapped by putting

```
        bvs.s     invalid
```

after each asl and add instruction in "mul_10" and also after the dbra instruction at "try_next". Also, strictly it should appear as well after the "addq.l #1,d5" at "make_even" to trap the case where a user has entered the number 2,147,483,647 (2^31 - 1) which would otherwise round up to -2^31.
The second comment relates to "get_text" on page 29. Here the address of "input" in A1 is put on the stack and later recalled. It would be shorter and quicker to replace "move.l (a7)+,a1" by "lea input,a1" and delete the initial "move.l a1,-(a7)".
The third comment is that the first instruction in "job_error" is not needed.
The fourth comment is that it is not necessary to close the channel just before "job_end" since the job's death will automatically do that.
The fifth comment is on "flush" on page 26. Obviously Norman found the need to flush the file buffers for safety. This is just the sort of addition needed to make a program robust in any circumstance and I hope that programmers will take note of this. I certainly now fear some of my programs may be in danger, since I have omitted such a step in them.

More from George in the next issue...

The Cover Disk

This cover disk has three programs, two of which are related to articles in this issue of QL Today. Run the boot program on the disk and follow the simple instructions to choose the programs you wish to expand (please see note on page 2 regarding UNZIP).

1: SAMPLER
This contains the Dilwyn Jones' program BMP that is described in the article "Perfect Partners". It contains a large number of bmp images that can be used as wallpaper. They are a sample of a full collection that will be available on DVD or set of CDs.

2: POpup eXtended
This is a software package that complements the "Poxology" article by Per Witte.

3: QL2PC full version
This is a full working version (i.e. not a demo) of the Just Words! flagship program QL-2-PC Transfer. The manual is to be found in text form in manual_txt and as a pdf file in manual_pdf

Where next?

by Geoff Wicks

A deserted hall at the recent Byfleet show. In practice it was not as bad as it looks, but less than thirty people were present. In spite of the low attendance there was a good atmosphere, but in both attendance and sphere Byfleet reflects what has happened at all UK shows this year. Attendance at Hove was slightly higher, but Manchester was far worse. In total probably no more than 40 QL-ers have attended a UK show this year.

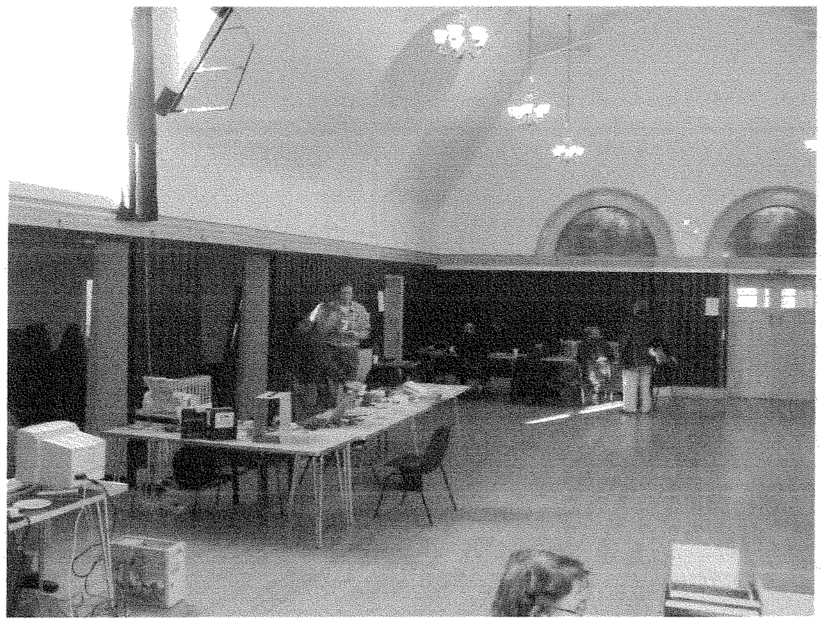
At present UK QL statistics make grim reading. In 2003 Quanta lost 18 members. In 2004 this rose to 33. In 2005 it was a staggering 53 members. In two years Quanta has lost somewhere between a quarter and a third of its membership. Statistics may give a picture of an ailing organisation, but Quanta has made some progress in recent months. The Quanta Magazine has improved considerably; there has been a welcome improvement in publicising committee decisions; and a greater willingness to exploit some of its capital.

Quanta is probably no longer an accurate barometer of the UK QL scene, and now faces important decisions on how to serve its members as a much smaller organisation.

It is even less representative of international QL activity. When you look back through recent issues of QL Today, you realise how active we QL-ers still are. SMSQ/E is continually being developed; there has been new graphics software; GD2 colours are no longer just for the experts, but are becoming more mainstream; and people have taken up the challenge to stretch EasyPtr programming to its limits.

Although QL Today has the occasional scare over a shortage of copy, we remain editorially highly viable. Over the year a steady stream of contributions has come in and, encouragingly, they have been varied in subject matter and usually of high quality. Our circulation is also holding up well. The QL Today team would like to thank all our readers and contributors for their support.

The QL scene it is not a picture of total doom and gloom, but we have to be prepared for change, particularly in the UK. Quanta is rapidly reaching a point where change will be necessary for survival. Smaller does not necessarily mean inferior. But then in what direction should Quanta now go?



Next time, as part of our full report on the Byfleet show, we shall be looking at the consequences for Quanta and future UK workshops. We would welcome the views of our readers. What do you see as the future for Quanta in general and UK workshops in particular?

Byts of Wood

by Roy Wood

We seem to have a few hardy perennial themes which will crop up at regular intervals in the QL community. Just recently three of these have been pointing their spiky head parapetwise.

The Hardware Debate

The first of these is the great hardware debate. On the one hand a bunch of die hard QLers (and I am not poking fun here at all) insist that the QL-hardware platform is by far superior to any kind of emulation and on the other, the emulation users point out the advantages of having multiple systems on one machine. I used to be firmly in the original hardware camp when I first started to use the QL - but that may have been because there were no emulators available. Even at the start of QBranch, in 1994, I was very much into the hardware side of things and that was the reason that I organised the two batches of Super Gold Cards that we sold and got so involved with the Q40. It was also the reason I was so disappointed when the Q40 business went so sour.

I believed right through that we should have a thriving hardware platform but there were signs, even in the late 90's that this was becoming more and more of a pipedream. Modern computer hardware has made such enormous strides and seen such drastic price drops that it is scarcely viable to use anything other than PC hardware. PC's are now available in supermarkets at incredibly low prices and with specifications that far exceed the 'high end' machines of four years ago. Disk drive capacities have soared, CPUs have got faster, graphics have got more powerful and removable USB memory is really very cheap.

Back in the early nineties, when I first thought of starting QBranch, the PC hardware and the QL were on a much more equal footing and, in many ways, the early QL hardware expansions made it a more viable machine, but it was not to be so we find ourselves at this juncture.

There are still many people who feel that there is enough demand out there for more QL hardware. All I can say to this is that there is not enough demand to make it worthwhile. By that I do not mean that there is not enough demand to make a profit but that there is not enough demand for the project to even break even on costs.

Any new hardware project has to take many things into account. Leave aside the development stages and the time and cost involved in even getting to a position where you have a rough sketch of what shape your new device is going to be. That whole area is going to be a throwaway as far as any QL device is concerned. Let us look at the circuit board, the basic PCB. In order to make these there are inherent costs in tooling. The company that is going to manufacture these for you will need to convert your blueprints into hard reality and that involves similar stages to that undertaken by printers in that a template has to be made, either physically or electronically, so that the board can be produced from it. There have been advances in this technique since I made the last batch of SuperGold Cards but for every step which reduces physical labour in favour of electronic or mechanised manufacture costs either even out or increase. This is just not an area for small time manufacture. Stuart Honeyballl used to estimate the lowest production run for a PCB to 50 units but I would say it was probably higher today.

Then we have the components. Another area where there are potential pitfalls and high costs. You can take the Q40 approach and buy in the parts as you need them but this leaves you open to shortfalls when the market dries up or, even worse, product variation which can lead to inconsistencies in the functions of the components. Some of the chips I bought for the Supergold Card made it not work because the manufacturer had changed the design and timing slightly but not changed the part number. Worse still CPUs and other components from different batches can have slightly different behaviour as Tony Tebby showed in Paris, when the original Colour Drivers first emerged for the Q40.

Most big companies get around this by building in all sorts of safeguards and by programming the devices well in the middle of the published specs for the device itself. They also have the advantage of getting the samples early and being able to order large quantities of the chips. This gives them a built in safety area. In our small world we have none of this. When the spec on come of the SuperGold Card chips changed we learned that the timing was just off the edge of the published range of the chip. The new version adhered tighter to the original spec and our process fell outside of its operational range. A bigger company would have raised hell about this and may have been able to get the manufacturer to change the design but it took me 6 months of shouting and cajoling to get my money back.

Where are you now?

As if all the foregoing was not bad enough we have the big problem of who is actually going to do this design/prototyping/manufacturing. Our best hope, and someone who has already provided a chunk of the best devices for QLs was Nasta and he has not really been heard of for a while.

There is still, however, an ongoing interest and desire for QL Hardware. I don't often get any Super Gold Cards, or Gold Cards for that matter, but when I do they get sold quickly. I am not sure though that I could sell a run of 50 quickly enough to make it worth the expense of building them. The same is true of most QL Hardware. While it is nice to be able to run stuff on an original machine that would be unbearably slow and not have much capacity. As soon as you add a Gold Card you are already running on a non original CPU. The same argument goes for the Aurora and, although you gain memory, some speed, and screen capability you do not improve the performance enough for it to really use the more recent software with any degree of efficiency.

This may sound like nit-picking to you but what I am saying here is 'Get real' Native hardware with a lot of memory and a fast CPU is just not going to happen. If you want the speed and memory to run all the latest advances in SMSQ/E and other software you will have to get an emulator and run it on another system, be that Windows, LINUX or even an Apple.

No maybe I have just annoyed someone enough to go out and make it just to prove me wrong - go on I dare you!

The QL On The Internet Question

The above comments apply just as well to the continuing cries for QL internet access. I can see that some people may only have a QL and may want to send and receive emails on it. This is possible, although I have not tried it myself. The only thing that having email on the QL would protect you from is Virus and Trojan attacks (they should really be called Greek attacks shouldn't they because it was the Greeks who gave the horse to the Trojans but there you are) The QL is never going to get to browse the web. Now that may be someone else I have annoyed. Same goes.

Format under SMSQ/E

Some time ago, several years ago in fact, I was at a QL show in Paris with my prototype Q 40. I asked Tony Tebby why SMSQ/E had stopped being able to format floppy disks. He said he was aware that it had and I demonstrated that I could format them under v 2.91 by just inserting a floppy disk and typing:

```
FORMAT FLP1_
```

at the command line. On both the Q40 and the Aurora based systems I was using at the time this was not working. On the Aurora it just locked up with the disk spinning for ages. Usually when the disk was removed and the system reset it was no longer readable. It seemed that the system was no longer able to recognise the difference between a DD and an HD disk. This was one of the problems at least. It was possible to format them by typing:

```
FORMAT 'FLP1_*D'
```

This problem is still with us and I still have to format my disks this way. Formatting from QPAC 2 does not work either. Funnily enough Disk Mate Five fares better because the interface allows you to select the disk density so I have done all my formatting from there in the last few years.

Just recently someone popped up on the Users List to mention this problem and there were quite a few replies saying that they also had these problems. Marcel looked back through the 'Changes' list and found that something in the floppy drive interface was changed around v2.81 but could not see what that was because his sources did not go back that far.

Tony Tebby was going to look into it but he was deep in the code for the colour drivers and I imagine he may have forgotten I mentioned it. What amazes me is that no-one else started shouting about this, after all formatting a floppy disk is something we all have to do. True the problem for me was worse because I had to format DD floppies for QL Today but surely others had the problem too. Maybe we should all report errors quicker if we want a better system.

Brain Teasing Time

One great thing about being as involved with things as I am is that I get sent all sorts of new test versions of things and a lot of the comments about these find their way into this column. This can, however work against you. When Wolfgang brought out the first versions of the Home Directory extensions I added them to my system.

I don't think I had any constructive things to add to the debate about them but it was fun to try them out. Just recently I received a new version of Suqcess for Bob Spelten and that seemed to work fine from QPAC2 but fail when run from the QDT menus. I immediately suspected a lack of integration with the Home Directory extensions in QDT and began to quiz Bob about this. After a few emails it seemed it all worked OK for him and not for me. I was in the process of writing the last part of the articles on BOOT files so I was looking at my own BOOT file and I realised that, rather then using the latest version of the Home Directory that is now part of SMSQ/E I was reloading an early Beta test version in the BOOT file and overwriting the new one. Once I removed that it all worked OK.
It was not made any easier by Per Witte who was having his own problems with the Home Directory. I noticed that he was mentioning how it did not work for him in the latest versions of SMSQ/E and I wondered if we were both seeing the same problem. Before I could write to him about it, however, he had worked out it was a corrupt FileInfo2 file causing his grief and that set me off looking for that before I discovered the true cause. Software testing is never straightforward at all.

Apology

I must publicly apologise to any of QBranch's customers who have been waiting a while for their orders in the last couple of months. I have had a lot of family problems, mostly to do with my elder relatives and then my wife managed to compound it all by falling down the stairs and breaking her foot. As a result of all of the problems I have had little time to deal with orders and questions of a QL nature. I am now getting back into it all and I hope to get everything back on track soon. Thank you all for your patience.

You Thought You'd Got Away with it but.......

We have been running a little late with the issues of QL Today so this will probably be the last issue before Christmas is upon us. Much as I hate the tinsel and snowflakes appearing in the shops while the Autumn sun is still warm I will rake this opportunity to wish you all a Good Christmas and a Happy New Year.
That said you can't escape my seasonal bad humour so, with a flourish, I would like to present, a little out of season, the annual Christmas Entertainment Listing for computer users.

Films and Music for Christmas

Just to prove he is not all Just Words our editor will star in a remake of

The (Geoff) Wickerman

Marcel Kilgus will also be in a remake of that famous black and white film police film, The Thin Blue Line' now retitled ***Queue PCs***

The Quanta Committee will start in the second League of Extraordinary Gentlemen film which will tackle the controversial claims about Jesus but will be even more absurd and hard to understand than the book and the original film ***The Dada Vinci Code***

LINUX fans will love ***Honey I shrunk the Kernel***

A Bond film where the villan burns himself on a QL heatsink ***'Goldcard Finger'***

A Triple Bill at the ICA film theatre in London on a theme of a malfunctioning copier will consist of: ***'A Scanner Darkly'***, ***'Click'*** and ***'Crash'***

For you retro boys out there we have ***The Black QLhlia, Black to the Future*** and ***Jane Err(or code)***

On a musical note we have another compilation

Jim Hunkins will be singing ***'QDT for Two'***

Dilwyn Jones will perform ***ZIPededodah***

Ron Dunnett will do his version of the Peter Paul and Mary hit: ***'Leaving on a Backplane'***

Jochen Merz will sing ***'A Few Of My Favourite THINGS'***

Nasta will do a reggae version of ***'The Liona Auroras Tonite-a'***

Wolfgang Lenerz will tackle that great 10 Years After Song, ***'I'm Going Home'***

Tony Firshman will sing ***'Tring Went The $ Of My Heart'***

The Van Der Auwera bothers will revive that Johnny Cache Hit, ***'I walk the LINEdesign'***

and the Mini Album will be rounded off with Marcel duetting with Jim on the Sonny And Cher hit, ***'The Beta Goes On'***

And as they say in Warner Brothers land, That's All Folks, Have a good Christmas.

... and please reserve the weekend 14th/15th of April for the Hove Show and possibly Quanta AGM - this is very last second news and will be confirmed next issue, on the J-M-S homepage etc.

Last Minute: QPC and SMSQ/E News

by Marcel Kilgus

It's been a while, over 6 months actually, since the last releases of both QPC and SMSQ/E, but in time for the Eindhoven meeting the job was done: QPC v3.33 and SMSQ/E v3.13 finally got released.

The big news regarding QPC is its much enhanced emulation core. It all started with George Gwilt wanting a 68020+ compatible QPC, as much of his software makes use of the extented instruction set these processors provide. He even volunteered to write it himself. So, after much delay on my part as I wanted to clean up the code first (some parts were over 10 years old and still commented in German) and needed to create a built environment George can easily use, he got the sources and created a new QPC version with many extended instructions.

Following this I reviewed the new code, cleaned it up a bit, added the remaining parts needed for full 68020 compatibility, created test applications, did regression tests against the old core and a QXL card donated to me and finally started a months long beta test for this crucial part of QPC. Truly a lot of work, but on the other hand not a single bug in the emulation has been found so far!

Regarding SMSQ/E, apart from numerous bug fixes mainly in the SBASIC core, there have been 2 major changes. A new job switching behaviour and changes to help speed up boot time.

The default CTRL+C behaviour was always very annoying to me: when pressing the keys the bottom-most window is picked to the top. The problem with this method is that the bottom-most window is often at that place for a good reason: you just don't currently need it. More often one wants to switch between the 2 or more top-most windows. With the new release you can just do that. When you hit CTRL+C not the bottom-most but the window just below the one you've currently worked with will be picked.

Now using this method one can already switch between the two top-most windows, but how can one get deeper than that? Easy, if you keep pressing the CTRL key and only press and release C, the picker will work its way to the bottom. But as soon as you release the CTRL key the game will again begin at the top! This might sound a bit complicated, but is really easy once you've played around with it for a bit. Also, if you really don't like the new behaviour it can be disabled using MenuConfig. All this is at least true for QPC, however I'm not sure how the other platforms currently handle the new functionality.

The second improvement is the wait time for all commands that start a new job. Traditionally commands like EX waited for half a second before returning to give the executed job a chance to open up a window. This for example ensures that buttons in the button frame show up in the order you start them in the boot file. Today's machines however are much faster and don't really need such a long pause anymore. To counter this problem there is now a new system variable ***sys_xdly*** (byte at ***$17e***) that determines the delay for a specific machine. This is preset to 5/50th of a second for QPC, which results in a much faster boot time. However for the full effect QPAC2 had to be updated too, and I did just that with release v1.45.

On machines that don't set sys_xdly half a second will again be used by default. To change sys_xdly yourself, for example to lower it even further to 3, you can poke a new value using this command:

```
POKE !;$17F,3
```

Finally there are a few new SBASIC commands, namely `YEAR%`, `MONTH%`, `DAY%` and `WEEKDAY%`. These take the same parameters as the DATE command and return the respective part of the date. `WEEKDAY%` returns `0` for Sunday, `1` for Monday etc.

Note from the Editor: These great news are really last-minute news ... 23rd of October! The new versions of SMSQ/E are available from the usual sources (J-M-S, Q Branch) either via mail (send in master disk and don't forget return postage rules) or through the J-M-S update site (QPAC2, SMSQ/E Goldcard...). The QPC2 Update is available at Marcels Homepage.

The QL Show Agenda

QL Meetings in Eindhoven

Saturday, 24th of March, 10:00 to 16:00
Pleincollege St. Joris, Roostenlaan 296

Thanks to the organiser, Sjef van de Molengraaf, the meetings at Eindhoven continue. Same venue as always (but a new, nice, large room straight on when you get into the main hall!) J-M-S will be there, as always. I am sure we will figure out on "international" meeting where the English dealers (and more international visitors) will attend. If it will be the March meeting, then QL Today will be the source for this information, as always.
Further meetings will be held in June and October.

HAPPY XMAS

and a

HAPPY NEW YEAR

from the QL Today team!

The Next Issue

We hope to have the next issue ready for you some time between middle or end of January.
As always, it depends on how quickly we will get reviews, articles etc.
Some articles will be continued in the next issues, we have some articles waiting which did not make it into this issue (58 pages AND cover disk!). But we need more material, as always. The more we get, the sooner we get it, the quicker the next issue will be in your hands.